Offline SLA-Constrained Deep Learning for 5G Networks Reliable and Dynamic End-to-End Slicing

Hatim Chergui, *Member, IEEE*, and Christos Verikoukis, *Senior Member, IEEE*

***Abstract*—In this paper, we address the issue of resource provisioning as an enabler for end-to-end dynamic slicing in software defined networking/network function virtualization (SDN/NFV)-based fifth generation (5G) networks. The different slices' tenants (i.e. logical operators) are dynamically allocated isolated portions of physical resource blocks (PRBs), baseband processing resources, backhaul capacity as well as data forwarding elements (DFE) and SDN controller connections. By invoking massive key performance indicators (KPIs) datasets stemming from a live cellular network endowed with traffic probes, we first introduce a low-complexity slices' traffics predictor based on a soft gated recurrent unit (GRU). We then build—at each virtual network function—joint multi-slice deep neural networks (DNNs) and train them to estimate the required resources based on the traffic per slice, while not violating two service level agreement (SLA), namely, *violation rate*-based SLA and *resource bounds*-based SLA. This is achieved by integrating dataset-dependent generalized non-convex constraints into the DNN offline optimization tasks that are solved via a non-zero sum two-player game strategy. In this respect, we highlight the role of the underlying hyperparameters in the trade-off between overprovisioning and slices' isolation. Finally, using reliability theory, we provide a closed-form analysis for the lower bound of the so-called *reliable convergence probability* and showcase the effect of the violation rate on it.**

***Index Terms*—5G, deep neural networks, dynamic slicing, non-convex optimization, reliability theory, SDN/NFV, SLA, violation rate.**

I. Introduction

NETWORK slicing is a key concept in 5G cellular systems. It yields the ability to run fully or partly isolated logical networks on the same physical network, offering thereby an increased statistical multiplexing [1]. Each logical network–or slice–is owned by e.g., an over-the-top (OTT) tenant (i.e., logical operator), and managed by the physical operator according to an established SLA. Nonetheless, the full isolation of slices at either the radio access or core network may have a high cost in terms of efficiency. Therefore, network slicing should be combined with solutions for dynamic orchestration of resources, at least at the network edge [2], [3]. In this context, the advent of the SDN/NFV paradigm is enabling the end-to-end virtualization and programmability of network functions, and paving the way to a flexible and dynamic resource allocation for the slices, which allows to exploit the available physical resources in a more efficient way [4], [5]. In this regard, machine learning (ML) techniques and in particular deep neural networks (DNNs) are expected to be the cornerstone in the automation of end-to-end resource provisioning. This includes schemes for traffic prediction such as long short-term memories (LSTMs) and gated recurrent units (GRUs) [6], to name a few. It also encompasses standard DNNs to model and estimate the required resources at each virtual network function (VNF) such as physical resource blocks at a transmission/reception point (TRP), radio resource connected (RRC) users' licenses at a virtual baseband processing unit (vBBU), enhanced radio bearers (ERAB) and signaling connections at a virtual DFE (vDFE) and virtual SDN controller (vSDNC), respectively. Nonetheless, such features are still in their early stage, as the resource management of current networks is mainly based on tweaked thresholds and hysteresis. In addition, devising low-complexity traffic prediction machine learning algorithms is an open issue in the literature. On the other hand, a notion of SLA is also required to properly convey network slices on top of a physical network, since this guarantees both slices' isolation and quality of service. In this intent, while we notice that some efforts have been deployed recently to assess the performance of provisioning algorithms in terms of SLA violation [7], there is no approach directly integrating the SLA constraints into the optimization of the DNN-based provisioning algorithms, given that this approach would enable to control the trade-off between slices isolation and resource dynamic allocation.

H. Chergui and C. Verikoukis are with CTTC, Barcelona, Spain. [e-mail: {hatim.chergui, cveri}@cttc.es].

A. Related Work

In [7] for instance, the authors point out that to realize the 5G network slicing, two complementary technologies are needed: (i) technical solutions that enable end-to-end network function virtualization (NFV), and provide the flexibility necessary for resource reallocation; and, (ii) data analytics that operate on mobile traffic measurement data, automatically identify demand patterns, and anticipate their future evolution. They then provide a convolutional neural network (CNN) to predict the traffic demand per slice. In this regard, we notice that this CNN strategy is of high complexity [8].

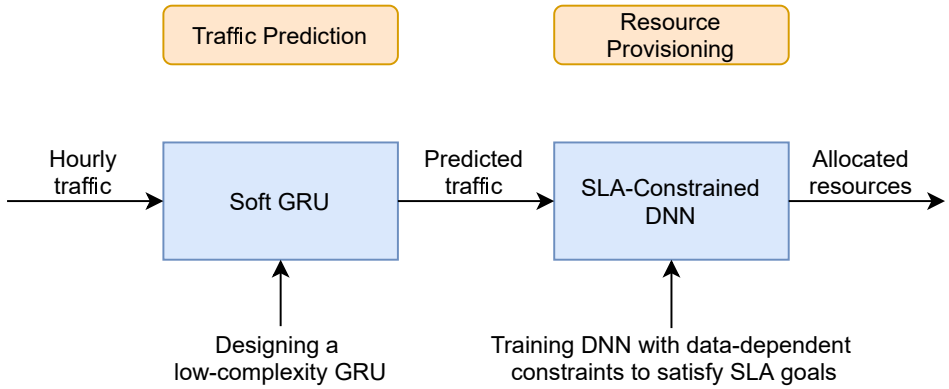


Figure 1: The main building blocks of the proposed data-driven network slicing resource allocation under SLA.

In [9], the authors use Holt-Winters forecasting procedure to analyze and predict future traffic requests associated to a particular network slice. Such a system, however, is hard to tune, scale and add exogenous variables [10].

Harnessing the exceptional feature extraction abilities of deep learning, [11] proposes a spatio-temporal neural Network (STN) architecture purposely designed for precise network-wide mobile traffic forecasting. It also presents a mechanism that fine-tunes the STN and enables its operation with only limited ground truth observations. The obtained traffic predictions, however, are not exactly matching the measured data.

In [12], based on a live network dataset by Telecom Italia [13], and adopting a service oriented network architecture with virtual functions rather than nodes, the authors present two machine learning approaches for control-plane traffic prediction in 5G networks, namely deep neural networks and recurrent neural networks (RNN), more specifically the so-called long short-term memories (LSTMs). The authors directly apply the standard LSTM without customizing it.

In an online setup, the authors in [14] have introduced a slice scheduler that allows existence of slices with bandwidth-based and resource-based reservations simultaneously, and implemented its prototype on a WiMAX testbed. The presented framework is intended only to rate optimization for slices' scheduling and cannot learn from global key performance indicators (KPIs) datasets to allocate various types of network resources.

B. Contributions

In this paper, we assume a SDN/NFV mobile architecture [15], wherein the traditional network components are smoothly evolved to comply with the virtualization and softwarization concepts. In this context, we investigate the following aspects as summarized in Fig. 1:

- Relying on live network OTT datasets, we first aggregate the OTTs traffics per slice. We then devise a custom low-complexity gated recurrent unit (GRU), called soft GRU, to predict the traffic for each slice.
- At each virtual function/interface of the SDN/NFV-based network, we build and train a joint multi-slice DNN model to estimate the resource[1] provision based on the traffic per slice. In this regard, we invoke live network key performance indicators (KPIs) datasets involving end-to-end metrics such as traffic volume per slice, downlink (DL) physical resource blocks (PRBs), CPU load and RRC connected users' licenses at the virtual baseband units (vBBUs), backhaul capacity, ERAB connections at the virtual data forwarding elements (vDFEs), and signaling connections at the virtual SDN controllers (vSDNs).
- Unlike existing online DNN optimization strategies, we introduce a new dataset-based training approach where the constrained DNN models are optimized for each slice to respect two types of SLA, namely, *violation rate*-based SLA and *resource bound*-based SLA. This is achieved by imposing dataset-dependent custom non-convex constraints to the DNN output and using a two-player non-zero sum game strategy to solve the resulting offline optimization task. In this intent, the SLA thresholds act as hyperparameters that can be fine-tuned by the infrastructure operator according to the SLAs with the slices' tenants. Note that we have adopted deep learning since it enables automatic discovery of important features from raw datasets, as well as yields generalized models, which is suitable for heterogeneous resources allocation.
- Based on reliability theory, we provide a closed-form analysis of what we call *reliable convergence probability*, where both the respect of SLA and convergence rate of the DNN models are jointly characterized, while highlighting the underlying trade-offs.

[1]The term resource encompasses physical, computational and licensing resources, depending on the corresponding network function.

C. Notations

We summarize the notations used throughout the paper in Table I.

Table I: Notations

Notation	Meaning
$\sigma(\cdot)$	Sigmoid function
$\pi(\cdot)$	Softplus function
$x_{t,n}$	GRU input data at time t for slice n
$\tilde{x}_{t,n}$	GRU candidate input at time t for slice n
$h_{t,n}$	GRU history signal at time t for slice n
z_t	GRU forget gate at time t
L	Number of neural network layers
N_l	Number of neurons at layer l
N_{B}	Batch size
$\ell(\cdot)$	Loss function
$\mathbf{W}_n$	Neural network weight for slice n
$\mathbf{b}_n$	Neural network bias for slice n
$\mathbf{s}_n$	Input features
$r_{m,n,k}$	Resource k at VNF m for slice n
$\alpha_{m,n,k}$	Lower-bound of resource k at VNF m for slice n
$\beta_{m,n,k}$	Lower-bound of resource k at VNF m for slice n
$\rho_{m,n,k}$	Target SLA violation rate for resource k at VNF m for slice n
$\lambda_{(\cdot)}$	Lagrange multipliers
R	Lagrange multiplier radius
$\mathcal{L}_{(\cdot)}$	Lagrangian with respect to $(\cdot)$

II. Network Architecture and Datasets

As depicted in Fig. 2, we consider a fully SDN/NFV architecture [15] wherein the baseband processing units run as softwarized virtual entities called vBBUs on datacenters close to the transmission/reception points (TRPs). On the other hand, all conventional enhanced packet core (EPC) entities no longer exist or are collapsed. Instead, the user plane packet gateways (PGWs) are replaced by virtualized data forwarding entities (vDFEs), while control plane serving gateway (SGW) and mobility management entity (MME) are replaced by a set of software applications implemented on top of a virtualized SDN controller (vSDNC) as suggested by many scientific research papers, e.g., [16], [17]. These applications could be newly defined or simply decomposed from functionalities of conventional EPC entities. For example, the MME and the SGW are traditionally sharing similar functionalities such as connectivity management, mobility management, while the MME and the home subscriber server (HSS) are sharing similar functionalities like authentication, attachment management. These functionalities can be formed or merged together as unified control elements or modules such as connectivity management (CM), mobility management (MM), and authentication management (AM). Note that the establishment of a slice consists on the end-to-end creation of dedicated VNFs (e.g., vBBU, vSDNC...).

A. Network Configuration

The collected KPIs correspond to an LTE-advanced (LTE-A) dense urban area, covered by 440 LTE-A eNodeBs (eNBs) and 3200 cells, including 800 MHz, 1800 MHz and 2.6 GHz bands. As the measurement data used in this work stems from an LTE-A live network that is not supporting the SDN/NFV framework yet, we summarize in Table II the necessary assumptions we have made throughout this paper to be aligned with the SDN/NFV architecture; in particular to aggregate the traffic at the different datacenters. In this regard, the eNB and vBBU traffics are the sum of the corresponding TRPs individual traffics, while the vBBU datacenter traffic is the aggregation of the related vBBUs. The vDFE and vSDNC traffics represent the whole network traffic.

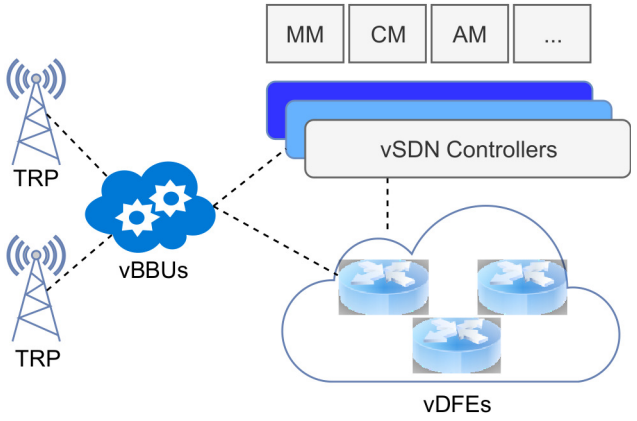


Figure 2: SDN/NFV-based network architecture.

Table II: Network Configuration

Entity	Quantity
TRP	3200
eNB	440
BBU datacenters	10 uniformly distributed, with $\times 100$ CPU resources compared to a single 4G eNodeB
DFE and SDN controller datacenters	1

B. Datasets

The measured datasets are based on two network components. First, thanks to their deep inspection capabilities, dedicated probes—usually installed at the core network—are collecting and analyzing the traffic per OTT at a granularity of 1 hour for each TRP. The traffic is then aggregated at eNB, vBBU datacenter and network levels for each OTT. Once the slices are defined, the traffic of the underlying OTTs is summed to yield the traffic per slice as depicted in Fig. 3. Second, the key performance indicators are collected by the operational support system (OSS) platform at TRP, eNB and network levels. The KPIs have a granularity of 1 hour and are formatted as detailed in Table III. Note that we have used Huawei's PRS tool to export the OSS KPIs (e.g., PRB usage, CPU load...) and Netscout of Tektronix to get the probes OTT KPIs.

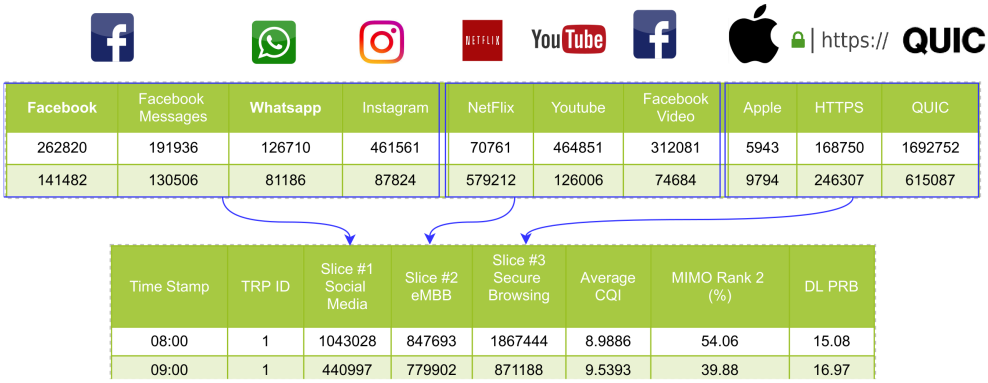


Figure 3: Slices creation and traffic aggregation at TRP level. Each row corresponds to a TRP at a given hour.

Table III: Datasets Features

TRP Feature	Description
OTT Traffics per TRP	Includes the hourly traffic for the top OTTs: Apple, Facebook, Facebook Messages, Facebook Video, Instagram, NetFlix, HTTPS, QUIC, Whatsapp, and Youtube
CQI	Channel quality indicator reflecting the average quality of the radio link of the TRP
MIMO Full-Rank	Usage of MIMO full-rank spatial multiplexing in %
DLPRB	Number of occupied downlink physical resource blocks
vBBU Feature	**Description**
OTT Traffics per eNB	Aggregated OTT traffics per eNB
CPU Load	CPU resource consumption in %
RRC Connected Users	Number of RRC users licenses consumed per eNB
Backhaul Feature	**Description**
OTT Traffics per BBU datacenter	Aggregated OTT traffics per BBU datacenter
Backhaul capacity	Effective aggregated throughput per BBU datacenter
vDFE/vSDNC Feature	**Description**
OTT Network Traffics	Aggregated OTT traffics over the network
ERAN Connections	Aggregated ERAB connections over the network
Signaling Connections	Aggregated signaling connections over the network

III. Soft GRU for Traffic Prediction

Let $x_{t,n}$ denote the traffic of slice n, $(n = 1, \ldots, N)$ at time t (in hours), and obtained by aggregating the corresponding OTTs' individual traffics. For instance, we assume that at a given VNF, eMBB slice's traffic is obtained by summing up the related hourly traffics of NetFlix, Youtube and Facebook Video. To ensure a proactive resource provisioning, we first need to predict the traffic volume in the next hour $t+1$, i.e., $\hat{y}_{t,n}$. In this intent, we introduce a new low-complexity gated recurrent unit (GRU) called *soft GRU* as depicted in Fig. 4. In contrast to the standard GRU, the proposed architecture involves only two gates, namely, an update gate that controls the contribution of the previous state and the current gate that yields the new input via a customized activation function π. While the light GRU initially introduced in [18], and lately simplified in [19], [20], relies on the simplification of the forget gate z_t or the batch normalization of the input data, the proposed soft GRU optimizes the generation of the candidate input $\tilde{x}_{t,n}$ by suppressing the history signal $h_{t-1,n}$ while introducing the *softplus* activation function to stabilize the obtained result, without changing the forget gate or preprocessing the dataset. The main building blocks of the soft GRU are formulated as follows:

$$h_{t,n} = (1 - z_t) \odot h_{t-1,n} + z_t \odot \tilde{x}_{t,n} \tag{1a}$$

$$\tilde{x}_{t,n} = \pi \left(W_x x_{t,n} + b_x \right) \tag{1b}$$

$$z_t = \sigma \left(W_z x_{t,n} + U_z h_{t-1,n} + b_z \right) \tag{1c}$$

where $\pi(x) = \log(1 + e^x)$ is the *softplus* function and σ is the sigmoid function. W_x, W_z and U_z stand for the GRU weights, while b_x and b_z represent the corresponding biases. The GRU module is then followed by a dense neural network layer that yields the final predicted traffic at the tth hour for slice n, $\hat{y}_{t,n}$. To optimize the parameters of this customized GRU over a training dataset of length T, we adopt the mean squared error standard loss function wherein we introduce an additional hyperparameter ϵ:

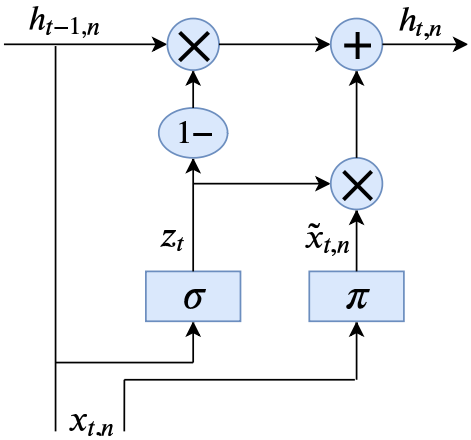


Figure 4: Soft GRU cell.

$$\ell_{\text{GRU}} = \frac{1}{2T}\sum_{t=1}^{T} \|\epsilon\hat{y}_{t,n} - y_{t,n}\|^2. \tag{2}$$

Indeed, The hyperparameter ϵ control the level of overprovisioning yield by the traffic prediction, and can be adjusted according to the operator resource provisioning strategy. The GRU training phase allows the determination of the optimal value of ϵ for an exact traffic prediction.

IV. End-to-End Resource Provisioning under SLA Constraints

In this section, we build deep learning models that, once fed with the predicted slices' traffics, enable to estimate the end-to-end required resources for each slice. Moreover, these models should—from the beginning—be trained in such a way to guarantee the respect of some target key performance indicators (KPIs) included in the slice's SLA. In practice, these (KPIs) turn out to be non-convex and result in a non-convex constrained deep learning exercise. In this regard, we consider for each slice n and virtual network function $m \in \{\text{TRP, vBBU, Backhaul, vDFE, vSDNC}\}$, a set of resources $r_{m,n,k}\,(k = 1,\ldots,K)$. Examples of resources are the DL PRBs at TRP and the CPU load at the vBBU datacenter. For notation simplicity and without loss of generality, we adopt neural networks of similar depth L wherefore the input features, weights and biases are denoted by $\mathbf{s}_n$, $\mathbf{W}_n$ and $\mathbf{b}_n$, respectively, while $\ell(\cdot)$ and N_{B} stand for the squared error loss function and the batch size, respectively. In the sequel, we formulate the deep learning-based resource provisioning problem under two types of non-convex SLA constraints, and show how one can proceed to solve the underlying optimization problems.

Note that each resource provisioning DNN model is unified multi-slice, i.e., jointly trained using the N slices' traffics and can be used to estimate the individual resources for each slice. This is achieved for a given slice n by keeping only the features related to that slice, and setting those corresponding to the remaining slices to zero.

A. Violation Rate-Based SLA

The advantage of this approach is that it overlooks the individual respect of SLA, and directly enforce an upper bound on the SLA violation rate, which is the common strategy followed by telecom operators. In this case, the deep learning training amounts to solving the optimization task expressed as,

$$\min \frac{1}{N_{\text{B}}}\sum_{i=1}^{N_{\text{B}}} \ell\left(r_{m,n,k}^{(i)}, \hat{r}_{m,n,k}^{(i)}\left(\mathbf{W}_n, \mathbf{b}_n, \mathbf{s}_n\right)\right), \tag{3a}$$

$$\text{s.t.}\ \mathbf{W}_{l,n} \in \mathbb{R}^{N_{l-1}\times N_l}, l = 1,\ldots,L+1, \tag{3b}$$

$$\mathbf{b}_{l,n} \in \mathbb{R}^{N_l \times 1}, l = 1,\ldots,L+1, \tag{3c}$$

$$\frac{1}{N_{\text{B}}}\sum_{i=1}^{N_{\text{B}}} \mathbb{1}\left(\hat{r}_{m,n,k}^{(i)} < \alpha_{m,n,k}\right) \leq \rho_{m,n,k}, \tag{3d}$$

$$\frac{1}{N_{\text{B}}}\sum_{i=1}^{N_{\text{B}}} \mathbb{1}\left(\hat{r}_{m,n,k}^{(i)} > \beta_{m,n,k}\right) \leq \rho_{m,n,k}, \tag{3e}$$

where $\mathbb{1}(\cdot)$ stands for the indicator function, and the constraint (3d) is imposing an upper bound on the SLA violation rate, i.e., the probability that the allocated resource $\hat{r}_{m,n,k}$ is outside the interval $[\alpha_{m,n,k}, \beta_{m,n,k}]$.

The loss function $\ell(\cdot)$ is a badly-behaving function of $\mathbf{W}_n$ because of the deep neural network structure, resulting in non-convex objective and constraint functions. In addition, the violation rate constraint is a linear combination of indicators, hence is not even subdifferentiable w.r.t. $\mathbf{W}_n$. Fixing this issue by replacing the constraints with differentiable surrogates introduces a new difficulty: solutions to the resulting problem will satisfy the surrogate constraints, rather than the actual ones. To sidestep this blocking point, let us consider the functions Φ_1 and Φ_2 defined as,

$$\Phi_1(\mathbf{W}_n) = \frac{1}{N_{\text{B}}}\sum_{i=1}^{N_{\text{B}}} \mathbb{1}\left(\hat{r}_{m,n,k}^{(i)} < \alpha_{m,n,k}\right) - \rho_{m,n,k}, \tag{4}$$

$$\Phi_2(\mathbf{W}_n) = \frac{1}{N_{\text{B}}}\sum_{i=1}^{N_{\text{B}}} \mathbb{1}\left(\hat{r}_{m,n,k}^{(i)} > \beta_{m,n,k}\right) - \rho_{m,n,k}, \tag{5}$$

and let Ψ_1 and Ψ_2 be sufficiently-smooth approximations of Φ [21] verifying

$$\Psi_1\left(\mathbf{W}_n\right) = \frac{1}{N_{\text{B}}}\sum_{i=1}^{N_{\text{B}}} \sigma\left(\alpha_{m,n,k}^{(i)} - \hat{r}_{m,n,k}\right) - \rho_{m,n,k} \leq 0, \tag{6}$$

$$\Psi_2\left(\mathbf{W}_n\right) = \frac{1}{N_{\text{B}}}\sum_{i=1}^{N_{\text{B}}} \sigma\left(\hat{r}_{m,n,k}^{(i)} - \beta_{m,n,k}\right) - \rho_{m,n,k} \leq 0, \tag{7}$$

where σ stands for the sigmoid function. The problem (3) can then be solved by invoking the so-called *proxy Lagrangian* framework [22]. This starts by forming two Lagrangians as follows:

$$\mathcal{L}_{\mathbf{W}_n} = \frac{1}{N_{\mathrm{B}}} \sum_{i=1}^{N_{\mathrm{B}}} \ell\left(\mathbf{r}_{m,n,k}^{(i)}, \hat{\mathbf{r}}_{m,n,k}^{(i)}\left(\mathbf{W}_n, \mathbf{b}_n, \mathbf{s}_n\right)\right) + \lambda_1 \Psi_1(\mathbf{W}_n) + \lambda_2 \Psi_2(\mathbf{W}_n), \tag{8a}$$

$$\mathcal{L}_{\lambda} = \lambda_1 \Phi_1\left(\mathbf{W}_n\right) + \lambda_2 \Phi_2\left(\mathbf{W}_n\right), \tag{8b}$$

where their optimization can be viewed as a non-zero-sum two-player game in which the $\mathbf{W}_n$-player wishes to minimize $\mathcal{L}_{\mathbf{W}_n}$, while the λ-player wishes to maximize $\mathcal{L}_{\lambda}$. Intuitively, the λ-player chooses how much to weigh the proxy constraint function, but does so in such a way as to satisfy the original constraint. By doing so, it reaches a nearly-optimal nearly-feasible solution to the original constrained problem. Note that $\lambda \leq R$, where R represents the maximum radius of Lagrange multipliers; introduced as a hyperparameter controlling the dependency to the constraints. In practice, we implement the deep learning objective function, the constraints (3d)-(3e) and the proxy constraints (6) and (7) on top of Google's `constrained optimization package` [23] that uses two different approaches to optimize the Lagrangians: a Bayesian optimization oracle for $\mathcal{L}_{\mathbf{W}_n}$ and projected gradient ascent for $\mathcal{L}_{\lambda}$. A definition of the oracle is given as follows:

Definition 1 (Approximate Bayesian Optimization Oracle)**.** *A δ-approximate Bayesian optimization oracle is a routine $\mathcal{O}_{\delta}$ that given a loss function/Lagrangian $\mathcal{L}$, returns the quasi-optimal weights $\mathbf{W}_n$ such that*

$$\mathcal{L}\left(\mathcal{O}_{\delta}\left(\mathcal{L}\right)\right) \leq \inf_{\mathbf{W}_n^{\star}} \mathcal{L}\left(\mathbf{W}_n^{\star}\right) + \delta. \tag{9}$$

B. Resource Allocation Bounds-Based SLA

To ensure slices' isolation, another type of SLA consists on thresholds imposed to the maximum and minimum resources granted by the deep learning model to each slice. Similarly to problem (6), we write this deep learning optimization task as follows:

$$\min \frac{1}{N_{\mathrm{B}}} \sum_{i=1}^{N_{\mathrm{B}}} \ell\left(r_{m,n,k}^{(i)}, \hat{r}_{m,n,k}^{(i)}\left(\mathbf{W}_n, \mathbf{b}_n, \mathbf{s}_n\right)\right), \tag{10a}$$

$$\text{s.t. } \mathbf{W}_{l,n} \in \mathbb{R}^{N_{l-1} \times N_l}, l = 1, \ldots, L+1, \tag{10b}$$

$$\mathbf{b}_{l,n} \in \mathbb{R}^{N_l \times 1}, l = 1, \ldots, L+1, \tag{10c}$$

$$\Phi_1 = \alpha_{m,n,k} - \min_i \hat{r}_{m,n,k}^{(i)} \leq 0, \tag{10d}$$

$$\Phi_2 = \max_i \hat{r}_{m,n,k}^{(i)} - \beta_{m,n,k} \leq 0. \tag{10e}$$

To construct the proxy constraints as done in problem (6), we seek smooth upper bounds on functions Φ_1 and Φ_2. In this regard, we invoke the smooth maximum and minimum functions expressed respectively as,

$$S_{\max}\left(\hat{r}_{m,n,k}^{(1)}, \ldots, \hat{r}_{m,n,k}^{(N_{\mathrm{B}})}\right) = \log\left(\sum_{i=1}^{N_{\mathrm{B}}} \exp\left\{\hat{r}_{m,n,k}^{(i)}\right\}\right), \tag{11}$$

$$S_{\min}\left(\hat{r}_{m,n,k}^{(1)}, \ldots, \hat{r}_{m,n,k}^{(N_{\mathrm{B}})}\right) = -\log\left(\sum_{i=1}^{N_{\mathrm{B}}} \exp\left\{-\hat{r}_{m,n,k}^{(i)}\right\}\right). \tag{12}$$

We then express the proxy constraints as,

$$\Psi_1 = \alpha_{m,n,k} - S_{\min} \leq 0, \tag{13a}$$

$$\Psi_2 = S_{\max} - \beta_{m,n,k} \leq 0. \tag{13b}$$

Finally, we form two Lagrangians,

$$\mathcal{L}_{\mathbf{W}_n} = \frac{1}{N_{\mathrm{B}}} \sum_{i=1}^{N_{\mathrm{B}}} \ell\left(\mathbf{r}_{m,n,k}^{(i)}, \hat{\mathbf{r}}_{m,n,k}^{(i)}\left(\mathbf{W}_n, \mathbf{b}_n, \mathbf{s}_n\right)\right) + \lambda_1 \Psi_1(\mathbf{W}_n) + \lambda_2 \Psi_2(\mathbf{W}_n), \tag{14a}$$

$$\mathcal{L}_{\lambda} = \lambda_1 \Phi_1(\mathbf{W}_n) + \lambda_2 \Phi_2(\mathbf{W}_n), \tag{14b}$$

and use the `constrained optimization package` [23] to optimize them similarly to the previous section.

V. Reliable Convergence Analysis

In this section, we analyze the convergence probability of the SLA-constrained deep learning models. To that end, we make use of reliability theory to account for the SLA violation effect. The following theorem provides a closed-form expression for the lower bound of the convergence probability, which unveils the effect of the underlying DNN hyperparameters such as the Lagrange multipliers radius, the error of the optimization oracle and the violation rate.

Theorem 1 (Convergence Analysis of the SLA-Constrained Neural Network)**.** *Consider that the deep neural network fails to fulfill the constraints with average violation rate $0 < \nu < 1$, and follows a geometric failure model. It is also assumed that $\mathcal{L}_{\mathbf{W}_n}$ is optimized using an oracle $\mathcal{O}_{\delta}$ with error δ, and let R and B_{Δ} stand for the Lagrange multipliers radius and the upper bound on the norm of subgradient $\nabla \mathcal{L}_{\lambda}$, respectively. Then, the reliable convergence probability satisfies,*

$$\mathbf{Pr}\left[\frac{1}{T_{\lambda}} \sum_{t=1}^{T_{\lambda}} \Big(\mathcal{L}_{\lambda}\left(\mathbf{W}_n^{(t)}, \lambda^{\star}\right) - \inf_{\mathbf{W}_n^{\star}} \mathcal{L}_{\lambda}\left(\mathbf{W}_n^{\star}, \lambda^{(t)}\right)\Big) < \varepsilon\right] \geq Q\left(\nu, \varepsilon\right), \tag{15}$$

where

$$Q\left(\nu, \varepsilon\right) = 1 - \frac{\nu}{1 + (\nu - 1)\exp\{-\varepsilon^2 / 2(2RB_{\Delta} + \delta)^2\}}. \tag{16}$$

Proof: First, by the subgradient inequality we have at time t,

$$\mathcal{L}_\lambda\left(\mathbf{W}_n^{(t)},\lambda^*\right)-\mathcal{L}_\lambda\left(\mathbf{W}_n^{(t)},\lambda^{(t)}\right)\leq\langle\nabla\mathcal{L}_\lambda^{(t)},\lambda^*-\lambda^{(t)}\rangle. \quad (17)$$

By invoking Holder's inequality, we get

$$\begin{aligned}\mathcal{L}_\lambda\left(\mathbf{W}_n^{(t)},\lambda^\star\right)-\mathcal{L}_\lambda\left(\mathbf{W}_n^{(t)},\lambda^{(t)}\right)&\leq\left\|\nabla_\lambda^{(t)}\right\|\left\|\lambda^\star-\lambda^{(t)}\right\|\\&\leq 2RB_\Delta.\end{aligned} \quad (18)$$

Combining (18) with Definition 1, we obtain

$$\mathcal{U}^{(t)}=\mathcal{L}_\lambda\left(\mathbf{W}_n^{(t)},\lambda^\star\right)-\inf_{\mathbf{W}_n^\star}\mathcal{L}_\lambda\left(\mathbf{W}_n^\star,\lambda^{(t)}\right)\leq 2RB_\Delta+\delta. \quad (19)$$

By means of Hoeffding-Azuma's inequality [24], we have

$$\Pr\left[\frac{1}{T_\lambda}\sum_{t=1}^{T_\lambda}\mathcal{U}^{(t)}<\varepsilon\mid T_\lambda=k\right]\geq 1-\exp\left\{-\frac{k\varepsilon^2}{2\left(2RB_\Delta+\delta\right)^2}\right\}, \quad (20)$$

where we consider that the deep neural network is reliable, i.e., respecting the SLA up to and including time $T_\lambda=k$. Therefore, recalling the geometric failure probability mass function P_k given by,

$$P_k=\nu\left(1-\nu\right)^k, \quad (21)$$

and combining it with (20), yields

$$\begin{aligned}\Pr\left[\frac{1}{T_\lambda}\sum_{t=1}^{T_\lambda}\mathcal{U}^{(t)}<\varepsilon\right]&\geq\sum_{k=0}^{+\infty}\nu\left(1-\nu\right)^k\\&\times\left(1-\exp\left\{-\frac{k\varepsilon^2}{2\left(2RB_\Delta+\delta\right)^2}\right\}\right).\end{aligned} \quad (22)$$

Finally, after some algebraic manipulations and using the fact that $\nu<1$, we get the desired result as in (15) and (16). ■

VI. Numerical Results

A. Neural Network Settings

Throughout this paper, we consider deep neural networks of $L=2$ hidden layers with $N_1=256$ and $N_2=8$ neurons, respectively. We set the training epochs to 300 and the optimizer to `Adam` with learning rate 0.01. These parameters are set following extensive experiments and turn out to yield the best results. The training dataset size varies from one network function to another. Hence, at TRPs and vBBUs levels, $N_{\mathrm{TR}}=21417$ and $N_{\mathrm{TR}}=9681$ samples, respectively, with batch size $N_{\mathrm{B}}=100$. At the vDFE and vSDN controller, where the traffic variation is non-bursty, we settle for $N_{\mathrm{TR}}=129$ with batch size $N_{\mathrm{B}}=10$. On the other hand, the test dataset at each network function consists of the hourly traffics of OTTs for a period of 5 days, i.e., 128 samples. The slices' traffics are obtained by aggregating the corresponding OTTs' traffics. In this work, we consider three slices, namely, enhanced mobile broadband (eMBB), Social Media and Browsing as shown in Fig. 3. In both training and test datasets, features normalization is activated. For the sake of simplicity, we drop the indexes m,n,k, and use vectors α and β instead. These vectors encompass the resource bounds corresponding to the different slices at a given network function, and can be easily understood from the context.

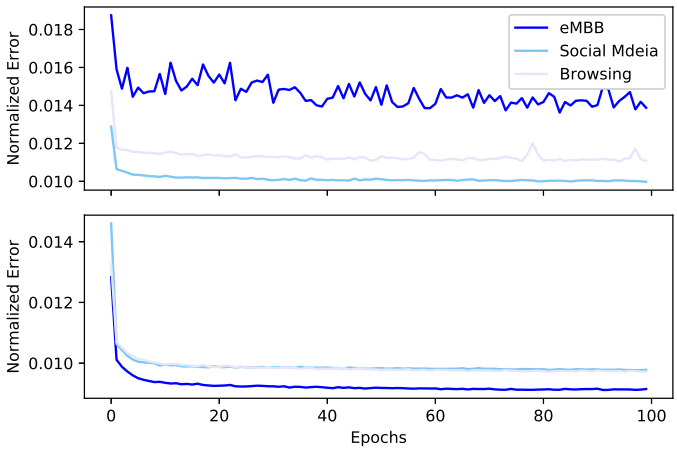


Figure 5: Normalized mean square error vs. epoch number for $\alpha=[15,0,5]$, and $\beta=[50,35,30]$ at TRP level, with $R=0.1$ and $R=0.9$, respectively.

B. Accuracy

To highlight the accuracy of the proposed DNN schemes, Fig. 5-(a) shows for instance that, as the number of iterations increases, the normalized training error of the joint multi-slice DNN model at TRP quickly decreases on average within few iterations, but keep fluctuating which increases slightly the algorithm convergence time. This behavior becomes accentuated in slices with tight resource bounds (e.g., eMBB and Browsing), and can be justified by the trade-off implied by the two-player game between the player minimizing the mean squared error and the one achieving the SLA constraints. In contrast, as depicted in Fig. 5-(b), when $R=0.9$, i.e., in case the constraints are quite omitted, the normalized mean squared error does not present any palpable fluctuations, and rapidly converges to lower levels compared to the first case, but at the expense of not fulfilling the SLA requirements.

C. Traffic Prediction Performance

Despite the presented GRU architecture is quite simple, it enables to track the traffic variation and yield concise predictions. The operator may fine-tune parameter ϵ to either overprovision or exactly match the required traffic per slice. A high value of ϵ results in underprovisioning while a small value leads to overprovisioning. The suitable value of ϵ can be determined at

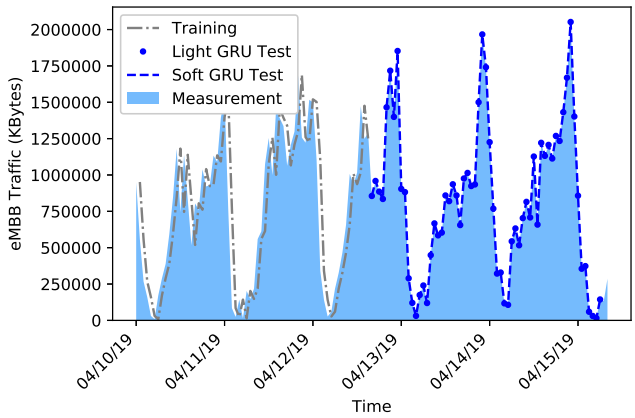


Figure 6: Predicted vs. measured traffics for eMBB slice at a TRP, with $\epsilon = 0.7$, 128×1 GRU.

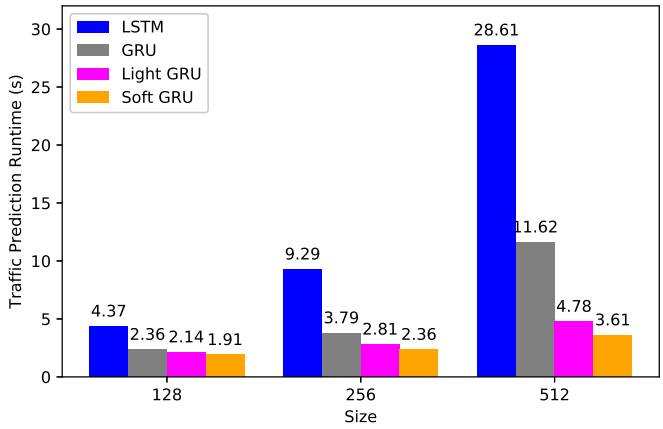


Figure 7: Prediction runtime for different recurrent networks architectures on a machine with Core i5-8500 CPU at 3 GHz and 16 GB RAM.

once via grid search by gradually changing it and comparing the training prediction with the ground truth. In this intent, a perfect match between the predicted and measured traffic volume is obtained for $\epsilon = 0.7$ and a GRU of size 128×1 as depicted in Fig. 5. Note that we run several training trials to find the optimal value of ϵ. We then use it to predict the traffic in a live evaluation dataset.

On the other hand, while we note a similar accuracy for both light GRU and soft GRU as shown in Fig. 6, we compare their time complexity along with other state-of-the-art (SoA) architectures like LSTM and standard GRU. In this case, we notice that our soft GRU presents the lowest runtime, especially when the number of needed GRU cells is high (e.g. 512) as depicted in Fig. 7.

D. Performance of Violation Rate-Based SLA

In this case, the deep learning models are optimized to respect the upper bound imposed to the SLA violation rate. As revealed by Fig. 8 and Fig. 9, we study the variation of the actual violation rate with respect to two hyperparameters, namely, the Lagrange multipliers radius R, and the upper bound ρ. In this regard, we recall that a small value of R lead to small multipliers λ_1 and λ_2 in (14), and the effect of the constraints becomes accentuated. On the other hand, ρ is the target violation rate threshold that the DNN output should respect with an acceptable probability.

In Fig. 8, we first remark that the actual violation rate is highly sensitive to the variation in R and ρ, which is not the case in Fig. 9 where the obtained violation rate is less sensitive to the hyperparameters. This behavior is due to the bounds α and β, wherefore their large difference (100 Mbps in the backhaul case) reduce the probability of violating the bounds and thereby results in a low sensitivity. This property is interesting from a network optimization viewpoint, given that wherever the number of resources is limited—like the DL PRBs—we should adopt the minimum setting of R and ρ to ensure the lowest violation rate, while in the case of relatively abundant resources—such as in the backhaul—the inter-slice isolation is easier and we may relax the constraints by tolerating fair values for R and ρ.

On the other hand, with low Lagrange multiplier radius $R = 0.1$, the DNNs model the provision of the required resources while respecting the target violation threshold ρ as depicted in Fig. 8 and Fig. 9. By increasing R, the problem (3) becomes unconstrained, and therefore breaches the maximum violation threshold ρ in some cases. Moreover, by increasing ρ, the DNN models are relaxed and the incurred violation rate is higher. Therefore, we conclude that, in practice, the infrastructure operator may adopt a dynamic parameter fine-tuning, where during busy hours—when a conflict between the slices is expected—one set $R = 0.1$ and at quiet times one set $R = 0.9$.

E. Performance of Resource Bounds-Based SLA

In this scenario, we impose bounds on the allocated resources at each network function. We start by showcasing the resource allocation results for SoA unconstrained DNN, and according to Fig. 10, it turns out that the target resource bounds are not respected as shown in the histogram distribution, since the DNN model has been trained without constraints in this case. In contrast, at TRP level, for example, when the constraints of problem (10) are active, i.e., when $R = 0.1$, the number of assigned DL PRBs to eMBB and Social Media slices are higher than 15 and 5 DL PRBs, respectively, as shown in Fig 11-(a). When $R = 0.9$, the lower bound α, for instance, is not taken into account as depicted in Fig. 11-(b). A more insightful representation is given by the histograms in Fig. 11, where we

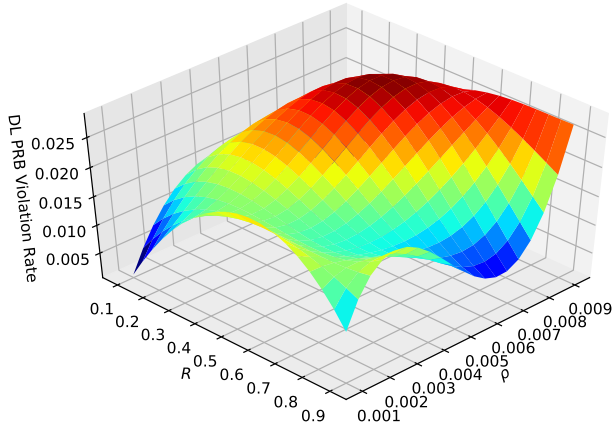


Figure 8: DL PRB violation rate vs. R and ρ for eMBB slice, with $\alpha = 1$ and $\beta = 30$ PRBs.

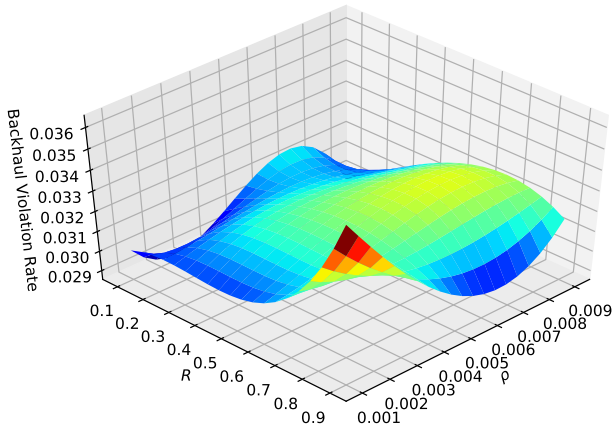


Figure 9: Backhaul capacity violation rate vs. R and ρ for Social Media slice, with $\alpha = 250$ Mbps and $\beta = 350$ Mbps.

easily identify the effect of the imposed SLA on the number of allocated DL PRBs. Indeed, with $R = 0.1$, most of eMBB PRBs grants are higher than 15 DL PRBs, while with $R = 0.9$ there is approximately 2300 samples below 10 PRBs.

On the other hand, we remark that the resource provisioning follows the same trend as the traffic since the latter serves as input to the DNN models. Hence, in Fig. 12 and Fig. 13, we show the CPU consumption and RRC connected users per slice for a single vBBU instance, and verify that the SLA is respected for the three slices, since $R = 0.1$. It can be seen that the number of RRC connected users for eMBB slice is lower than Social Media slice that is viewed as a massive access service. We also note that the presented CPU consumption and RRC connected users are with respect to a single vBBU instance that is processing the data of one eNB.

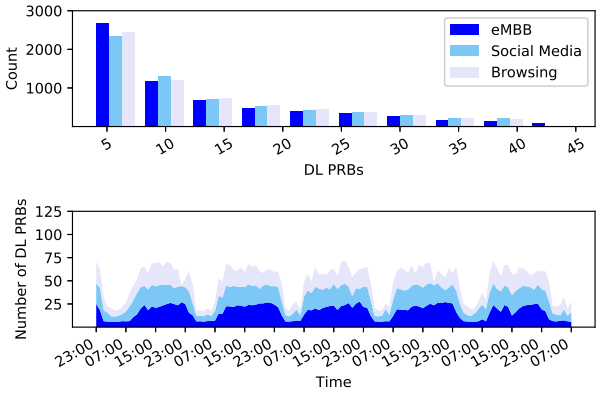
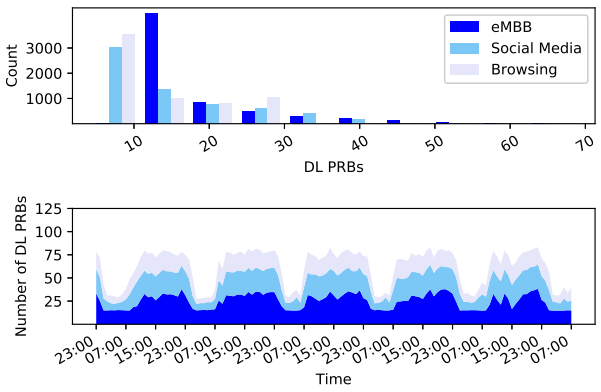
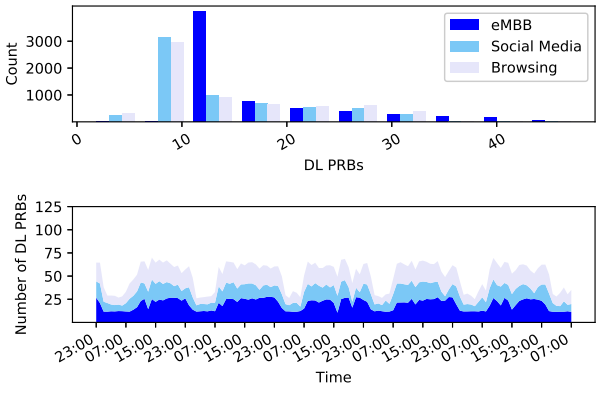


Figure 10: DL PRBs evolution and distribution per slice for SoA unconstrained resource allocation.



(a) $R = 0.1$



(b) $R = 0.5$

Figure 11: DL PRBs evolution and distribution per slice for resource bounds SLA, with $\alpha = [15, 0, 5]$ and $\beta = [50, 35, 30]$ PRBs.

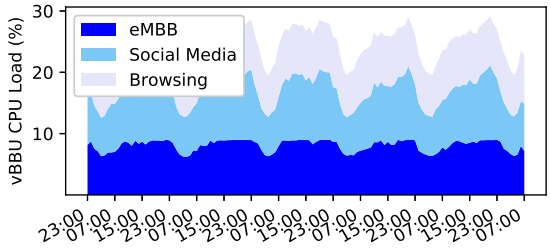


Figure 12: CPU consumption per slice for a single vBBU instance in resource bound SLA setting with $\alpha = [0\%, 0\%, 0\%]$, $\beta = [10\%, 15\%, 20\%]$, and $R = 0.1$.

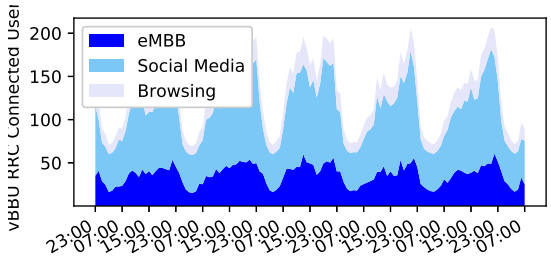


Figure 13: RRC connected users licenses per slice for a single vBBU instance in resource bound SLA setting with $\alpha = [0, 50, 0]$, $\beta = [75, 100, 100]$, and $R = 0.1$.

In addition, Fig. 14 depicts the backhaul capacity license granted to each slice for a single vBBU instance and under active SLA constraints. In this case, we can see that since the lower bound α for eMBB is 20 Mbps, the capacity thereof does not present a quiet time compared to Social Media and Browsing slices whose lower bounds are both at 0 Mbps. Imposing a lower bound might be seen as ensuring an isolation between the different slices, where even during low traffic periods a slice is allocated with a minimum number of resources. By tweaking the hyperparameter R, the physical operator may find the trade-off between overprovisioning and isolation, i.e., between following the traffic dynamics and fulfilling the *resource bounds* SLAs.

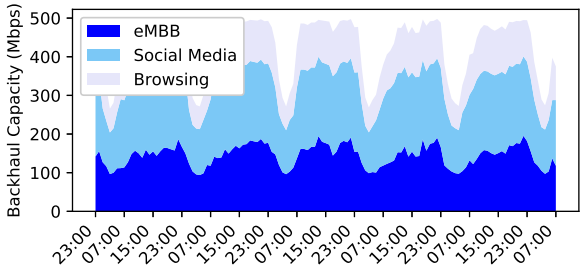


Figure 14: Backhaul capacity licenses per slice for a single vBBU instance with $\alpha = [50, 100, 50]$, $\beta = [150, 200, 150]$, and $R = 0.1$.

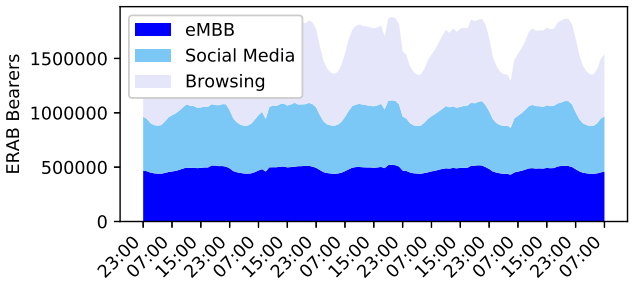


Figure 15: ERAB bearers per slice at the vDFE for $\alpha = [5 \times 10^5, 5 \times 10^5, 0]$, $\beta = [10^6, 10^6, 10^{10}]$, and $R = 0.1$.

Similarly, Fig. 15 and Fig. 16 present the assigned ERAB bearers and signaling connections at the vDFE and SDN controller, respectively. They are obtained by feeding the corresponding DNN models with the aggregated traffic over the whole network, i.e., the 440 eNBs. Given the imposed lower bounds as well as the fact that the eNBs have not the same busy and quiet hours, the network level ERAB bearers and signaling connections either present a slight quiet time like in Browsing and Social Media slices, or almost no quiet time like in eMBB slice. In all cases, thanks to these estimated dynamic resources per slice, the operator may efficiently manage the ERAB bearers and signaling connections licenses pools by avoiding dedicated static license distribution, which paves the way to operational expenditure (OPEX) savings while guaranteeing slices isolation.

F. Reliable Convergence

Fig. 17 depicts the lower bounds of the reliable convergence probability as a function of the regret ε. In this regard, $B_\Delta = 15.4$ is the practical maximum value of the gradient yield by the optimizer over the training dataset. As expected, a high violation rate ν leads to the decrease of $Q(\nu, \varepsilon)$. With a low violation rate $\nu = 0.01$ and $R = 0.1$, one can easily achieve a regret $\varepsilon = 0.1$ with probability $Q(\nu, \varepsilon) = 0.83$. From a design perspective, to achieve a low ν, the physical operator needs to agree reasonable resource bounds α and β with the slices' tenants.

VII. Conclusion

In this paper, we first present a low-complexity network slices' traffics predictor based on a soft gated recurrent unit (GRU), where some components have been dropped without impacting the performance. We then use the predicted traffics to feed several deep learning models trained offline to perform end-to-end dynamic and reliable resource slicing under dataset-dependent generalized non-convex SLA constraints. The concerned network resources are the DL PRBs at TRP, the CPU

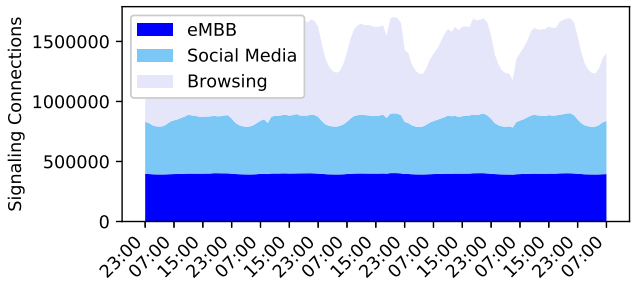


Figure 16: Signaling connections per slice at the vSDN controller for $\alpha = [5 \times 10^5, 5 \times 10^5, 0]$, $\beta = [10^6, 10^6, 10^{10}]$, and $R = 0.1$.

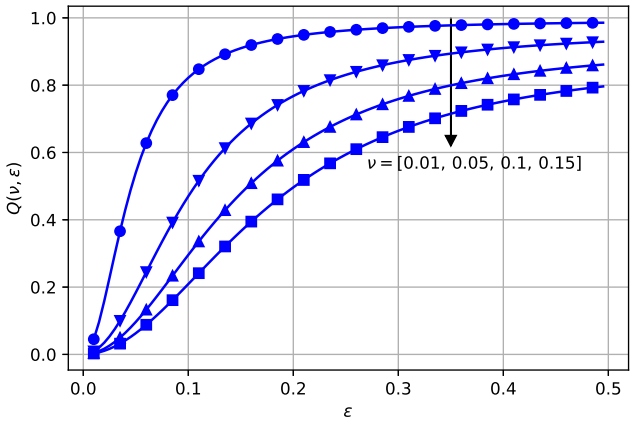


Figure 17: Reliable convergence probability vs. ε for $R = 0.1$, $B_\Delta = 15.4$ and various violation rates ν.

load and RRC connected users at vBBU datacenter, backhaul capacity, ERAB bearers at vDFE and signaling connections at vSDN. In this respect, we show that by properly tweaking the constraints' Lagrange multiplier radius, the physical operator may control the trade-off between resource overprovisioning and slices isolation. Finally, inspired by reliability theory, we introduce the concept of reliable convergence and derive a closed-form expression for the lower bound of the convergence probability. We also study the effect of the underlying hyperparameters, and provide some recommendations to ensure a fair SLA.

VIII. Acknowledgment

This work has been supported by the research project 5G-SOLUTIONS (856691).

References

[1] NGMN Alliance, "Description of network slicing concept," [Online]. Available: https://www.ngmn.org, accessed Mar. 2019.

[2] C. Marquez *et al.,* "How should I slice my network? A multi-service empirical evaluation of resource sharing efficiency," in *MobiCom'2018,* pp. 77-84, 2008.

[3] X. Foukas *et al.,* "Network slicing in 5G: Survey and challenges," in *IEEE Comm. Mag.,*, vol.55, no. 5, pp. 94-100, May 2017.

[4] Y. Zaki *et al.,*"LTE wireless virtualization and spectrum management," in *Wireless and Mobile Networking Conference (WMNC), 2010*, Third Joint IFIP, pp. 1â6, Oct 2010.

[5] M. Jiang *et al.,* "Network slicing management and prioritization in 5G mobile systems," in *European Wireless EW 2016*, Oulu, Finland, 2016.

[6] K. Cho *et al.,* "Learning phrase representations using RNN encoderâdecoder for statistical machine translation," in *EMNLP'2016*, Doha, Qatar, pp. 1724–1734, Oct. 2014.

[7] D. Bega *et al.,* "DeepCog: Cognitive network management in sliced 5G networks with deep learning," in *IEEE INFOCOM'2019,* Paris, France, May 2019.

[8] K. He and J. Sun, "Convolutional neural networks at constrained time cost," [Online]. Available: https://arxiv.org/pdf/1412.1710.pdf

[9] V. Sciancalepore *et al.,* "Mobile traffic forecasting for maximizing 5G network slicing resource utilization," in *IEEE INFOCOM'2017,* Atlanta, GA, USA, May 2017.

[10] N. Laptev *et al.,* "Time-series extreme event forecasting with neural networks at Uber," in *NIPS Time Series Workshop,* California, USA, Dec. 2017.

[11] C. Zhang and P. Patras, "Long-term mobile traffic forecasting using deep spatio-temporal neural networks," in *Mobihoc'2018,* pp. 231-240 , 2018.

[12] I. Alawe *et al.,* "Improving traffic forecasting for 5G core network scalability: A Machine Learning approach," in *IEEE Net. Mag.,* vol. 32, no. 6, pp. 42-49, Nov. 2018.

[13] T. Italia, "A multi-source dataset of urban life in the city of Milan and the province of Trentino dataverse." [Online]. Available: dx.doi.org/10.7910/dvn/EGZHFV

[14] R. Kokku *et al.,* "NVS: A substrate for virtualizing wireless resources in cellular networks," in *IEEE/ACM Trans. Net.,* vol. 20, no. 5, pp. 1333-1346, Oct. 2012.

[15] V.-G. Nguyen *et al.,* "SDN/NFV-based mobile packet core network architectures: A survey" in *IEEE Comm. Surveys Tutorials*, vol. 19, no. 3, pp. 1567-1602, Third-quarter 2017.

[16] R. Guerzoni *et al.,* "SDN-based architecture and procedures for 5G networks," in *1st Int. Conf. 5G Ubiquitous Connectivity (5GU),* 2014, pp. 209-214.

[17] R. Trivisonno *et al.,* "SDN-based 5G mobile networks: Architecture, functions, procedures and backward compatibility," in *Trans. Emerg. Telecommun. Technol.,* vol. 26, no. 1, pp. 82-92, 2015.

[18] G.-B. Zhou *et al.,* "Minimal gated unit for recurrent neural networks," in *Int. J. Autom. Comput.,* vol. 13, no. 3, pp. 226-234, 2016.

[19] J. Heck and M. S. Fathi, "Simplified minimal gated unit variations for recurrent neural networks," in *IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS),* Oct. 2017.

[20] M. Ravanelli *et al.,* "Light gated recurrent units for speech recognition," in *IEEE Transactions on Emerging Topics in Computing,* vol. 2, no. 2, pp. 92-102, Apr. 2018.

[21] J.A.K. Suykens *et al.*, *Advances in Learning Theory: Methods, Models and Applications,* IOS Press, May 2003.

[22] A. Cotter *et al.,* "Training well-generalizing classifiers for fairness metrics and other data-dependent constraints" [Online]. Available: arxiv.org/abs/1807.00028.

[23] A. Cotter *et al.,* Constrained Optimization (TFCO). [Online]. Available: https://codeload.github.com/google-research/tensorflow_constrained_optimization/zip/master

[24] A. Hoeffding, "Probability inequalities for sums of bounded random variables," in *Journal of the American Statistical Association,* vol. 58, no. 301, pp. 13-30, Mar. 1963.

Hatim Chergui (M'12) received the Bachelor's degree in telecommunications engineering from the Institut National des Postes et Télécommunications (INPT), Rabat, Morocco, in 2007, and the Ph.D. degree (with honors) in electrical engineering and telecommunications from Télécom-Bretagne, Brest, France, in 2015. Since 2008, he has been a radio network planning and optimization engineer, with extensive industry experience in providing 3G/4G consulting services to Huawei Technologies, Morocco. He has also served as a radio technologies expert at the Moroccan operator INWI, Casablanca, Morocco. He is currently a postdoctoral researcher at the Catalan Telecommunications Technology Center (CTTC) in Barcelona, Spain. His research interests lie in the area of performance analysis and machine learning applied to wireless communications.

Christos Verikoukis (SM'07) received the Ph.D. degree from the Technical University of Catalonia, Barcelona, Spain, in 2000, in the area of broadband indoor wireless communications.

He is currently a Fellow Researcher with Telecommunications Technological Centre of Catalonia (CTTC/CERCA), Castelldefels, Spain, and an Adjunct Professor with UB. He has authored more than 115 journal papers and more than 180 conference papers. He has coauthored more than three books, 14 chapters, and two patents. He has participated in more than 30 competitive projects and has served as the Principal Investigator of national projects. He has supervised 15 Ph.D. students and five postdoctoral researchers.

Dr. Verikoukis is the Chair of the IEEE Communications Society's Communications Systems Integration and Modeling Technical Committee. He received the Best Paper Award at the 2011 IEEE International Communications Conference, the IEEE GLOBECOM 2014 and 2015, and the 2016 European Conference on Networks and Communications, and the EURASIP 2013 Best Paper Award of the Journal on Advances in Signal Processing.