JOURNAL OF
Optical Communications
and Networking

Decentralizing machine-learning-based QoT estimation for sliceable optical networks

Tania Panayiotou,[1,*] Giannis Savva,[1] Ioannis Tomkos,[2] and Georgios Ellinas[1]

[1]Department of Electrical and Computer Engineering and KIOS Research and Innovation Center of Excellence, University of Cyprus, Nicosia, Cyprus
[2]Department of Electrical and Computer Engineering, University of Patras, Patras, Greece
*Corresponding author: panayiotou.tania@ucy.ac.cy

Received 9 January 2020; revised 5 March 2020; accepted 11 March 2020; published 16 April 2020 (Doc. ID 387853)

Dynamic network slicing has emerged as a promising and fundamental framework for meeting 5G's diverse use cases. As machine learning (ML) is expected to play a pivotal role in the efficient control and management of these networks, in this work, we examine the ML-based quality-of-transmission (QoT) estimation problem under the dynamic network slicing context, where each slice has to meet a different QoT requirement. Specifically, we examine ML-based QoT frameworks with the aim of finding QoT model/s that are fine-tuned according to the diverse QoT requirements. Centralized and distributed frameworks are examined and compared according to their model accuracy, routing and spectrum allocation (RSA) accuracy, and CPU (training time) and RAM (memory) requirements. We show that the distributed QoT models outperform the centralized QoT model in accuracy and CPU usage. The RSA accuracy, i.e., measuring the accuracy of the models with regard to the QoT-aware RSA decisions, is sufficiently high for both frameworks. Regarding the RAM usage, as the distributed framework has to train in parallel several QoT models, it may require higher memory, especially as the number of diverse QoT requirements increases. This memory, however, tends to be reserved for a shorter period of time. Moreover, this work develops a dynamic multi-slice QoT-aware (RSA) framework that integrates the ML-based QoT models. The aim is to examine the network performance when the diverse QoT models are considered, as opposed to the state-of-the-art single-slice QoT-aware RSA approach where all connections/slices are provisioned according to a single QoT requirement. We show that the multi-slice QoT-aware RSA approach significantly improves network performance, a clear indicator that the commonly considered single-slice QoT-aware RSA approach may lead to connection overprovisioning. © 2020 Optical Society of America

https://doi.org/10.1364/JOCN.387853

1. INTRODUCTION

Network slicing is considered today as a promising and fundamental framework for supporting 5G mobile networks and their emerging services (e.g., connected autonomous cars, connected robots) and applications (e.g., video streaming, augmented reality, virtual reality) with very diverse service level requirements (e.g., bit rates, bit-error rates (BERs), latency, availability) [1]. In general, network slicing is a technology allowing multiple virtual networks (slices) to be created on top of a common shared physical infrastructure, with the slices being customized to meet the specific needs of each service [1,2]. The realization of this service-oriented view of the network is based on the principles of software-defined networking (SDN) and network function virtualization (NFV), allowing for the dynamic (on-demand) creation of virtual slices that can also be linked through software. On this basis, many telecom operators (e.g., Ericsson, Nokia, and AT&T) have in the past few years adopted network slicing as an emerging technology and business model [2,3].

In general, the implementation of network slicing is conceived as an end-to-end feature spanning over different technology domains. While relevant work focuses mostly on the implementation of network slicing in the radio access network (RAN) segment [1,2,4,5], recent work also focuses on extending the scope of network slicing beyond the RAN to include both mobile and transport (optical) network segments [6–8]. In particular, recent work deals with SDN/NFV demonstrations, network orchestration implementations, and traffic demand analysis, towards the vision of end-to-end network slicing. In general, SDN/NFV, network orchestration, and analytics are considered as the key ingredients for successful implementation of end-to-end network slicing [1]. However, regarding the efficient control and management of these slices (i.e., enabling not only their creation but also their smart deployment at the right place, at the right time, with optimized resources), little has been done so far.

Without doubt, machine learning (ML) will be instrumental in the efficient control and management of these slices by enabling automation of the network planning functions. This is especially true in current networks, where network planning functions are becoming increasingly complex in an uncertain network environment that is continuously changing, supporting heterogeneous applications and services. Existing ML applications [9,10] focus on traffic demand predictions and resource allocation optimization [11–15], fault detection/localization [16–19], attack detection/identification [20,21], and quality-of-transmission (QoT) estimation [22–26]. In most of these works, however, the diverse optical service level agreements (OSLAs) of the next generation optical networks [27] are not specifically considered. Since the QoT requirement (i.e., BER) is among the OSLA specifications [27], in this work, we examine the ML-based QoT estimation problem for sliceable optical networks, where each slice has to meet a different QoT requirement. As the OSLAs are usually set *a priori* between the telecom operators and the services/applications, the diverse QoT requirements can also be known *a priori*, hence allowing for the application of ML for QoT model estimation. Note that the use of ML applications for QoT estimation aims at alleviating the drawbacks of the static Q-factor models traditionally used for the QoT estimation of unestablished connections [28–30] (i.e., lack of self-adaptiveness to network changes, highly overestimating the nonlinear physical layer impairments, underutilization of network resources [22,31]).

A. Our Contribution

The ML-based QoT estimation problem has been extensively studied in the literature. Commonly, the aim is to find a QoT model that is fine-tuned to the QoT requirement that best fits all the diverse QoT requirements. Thus, for ensuring sufficient QoT for all services (regardless of their actual QoT requirement), the QoT model has to accommodate the highest QoT requirement among all services (i.e., lowest BER). However, this practice may lead to connection overprovisioning. As optical networks are already transforming to a multi-slicing network planning scenario [6–8], the QoT estimation problem has to also be transformed accordingly. Towards this direction, our previous work [32], examined centralized and distributed ML-based QoT estimation frameworks with the objective of finding QoT estimation model/s that are fine-tuned to the diverse QoT requirements of each optical network slice.

Specifically, in [32], an SDN-based hybrid framework is examined that consists of both a centralized controller and several distributed controllers. In the centralized QoT framework, a multiclass classifier is trained at the central controller according to global network information (i.e., according to all QoT requirements). In the distributed QoT framework, a set of binary classifiers is trained in parallel and independently within a set of distributed controllers. Each distributed controller stores information that is relevant only to its own slice type, with each slice type being formed by the connections with the same QoT requirements. Hence, each distributed model is trained according to a different QoT requirement. The preliminary results of [32] showed that the distributed approach outperforms the centralized approach in model accuracy, accuracy per class, and training time, especially as the number of slice types increases. In this work, we significantly extend [32] as follows:

- We analytically describe the mathematical formulations of the proposed ML-based QoT estimation methods.
- We train both the centralized and distributed QoT models on new datasets consisting of more patterns and also considering different network loads.
- We train both the centralized and distributed QoT models according to neural network (NN) topologies consisting of different numbers of hidden layers and units. Specifically, as the centralized framework is based on a multiclass classification problem, which is in general harder to solve than its binary decompositions [33–35] (distributed QoT problem), the centralized framework is examined for a larger NN topology.
- The models/frameworks are compared not only according to their accuracy and CPU usage, but also according to their RAM usage and routing and spectrum allocation (RSA) accuracy.
- We develop dynamic multi-slice QoT-aware RSA heuristics integrating the diverse QoT models of both the centralized and distributed frameworks.
- The network's performance is compared when utilizing the proposed multi-slice QoT-aware RSA approach versus the case where the conventional single-slice QoT-aware RSA approach is used.

The rest of this work is organized as follows: Section 2 discusses related work, Section 3 provides the approach overview, while Section 4 provides the problem statement. The ML-based QoT estimation problem formulation is given in Section 5, and the multi-slice QoT-aware RSA heuristics are described in Section 6. Section 7 discusses the QoT models' training and evaluation, and Section 8 presents the network performance evaluation. Concluding remarks are given in Section 9.

2. RELATED WORK

ML applications for optical network planning have in the past few years attracted significant attention [9,10]. In their general form, existing ML applications assume an SDN-based optical network controller that centrally controls and manages the network [11,36] (Fig. 1). The central controller/orchestrator is equipped with storage, processing, and monitoring capabilities, and the ML applications run centrally according to the datasets found in the central knowledge database. Under this framework, the aim of the SDN-based controller is to efficiently manage the network resources so that the diverse quality-of-service (QoS) requirements of the different services are met, as closely as possible, by the virtual network topology (VNT). The VNT is in essence a single slice (virtual network) that has to accommodate as best as possible all the services.

Regarding the QoT estimation problem, which this work focuses on, an ML application runs centrally according to the monitored and stored lightpath information (i.e., BER

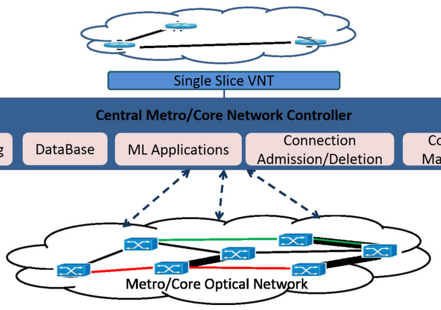
Single Slice VNT
Central Metro/Core Network Controller
Monitoring
DataBase
ML Applications
Connection Admission/Deletion
Connection Management
Metro/Core Optical Network

Fig. 1. Single-slice control and management framework.

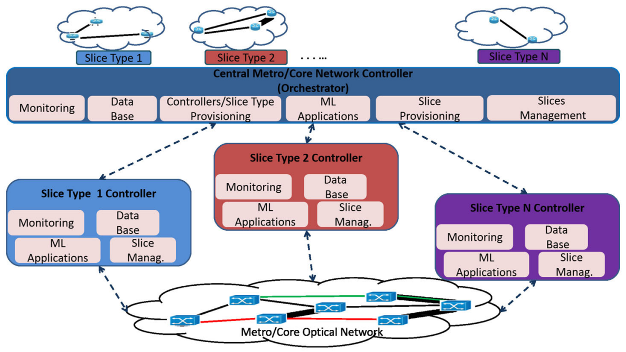

Fig. 2. Multi-slice control and management framework.

and lightpath's features). The objective is to find an accurate QoT estimation model that is subsequently used for centrally managing the network resources. In general, this model can be adapted to network changes by, for example, retraining the model according to information that is shifted in time. Under the single VNT assumption, however, the ML-based QoT model may lead to underutilization of network resources. Specifically, the ML-based QoT model, for best meeting the diverse QoT requirements of the various services, must be trained according to the highest QoT requirement (i.e., the lowest BER requirement) among all services. Therefore, the state-of-the-art problem is usually addressed as a binary classification problem that is based on a single BER threshold (i.e., the lowest BER requirement). The binary classifier is trained for estimating whether the BER of the candidate lightpath is above or below the threshold. Thus, the QoT model is capable of classifying the unestablished lightpaths into one of two classes: the infeasible QoT class or the feasible QoT class [22–24]. However, the practice of considering a single BER requirement may lead to connection overprovisioning, especially as the diversity of the various BER requirements increases.

To alleviate the connection overprovisioning effect, centralized and distributed ML-based frameworks are examined that specifically account for the diverse BER requirements of the slices. Both frameworks are evaluated and compared according to their model accuracy, RSA accuracy, CPU usage (training time), and RAM usage (memory requirement). A multi-slice QoT-aware RSA heuristic is then developed that integrates all the diverse QoT models. The single-slice QoT-aware RSA heuristic, integrating only the QoT model of the lowest BER requirement, is used as a benchmark. The reader should note that in the literature, existing QoT-aware RSA algorithms consider only a single BER requirement [28–30]. In this work, we show that the multi-slice QoT-aware RSA outperforms the state-of-the-art single-slice QoT-aware RSA approach by avoiding slice/connection overprovisioning, an indicator that the diverse BER requirements must be specifically taken into account during slice/connection provisioning.

3. APPROACH OVERVIEW

A sliceable optical metro/core network is assumed, in which connections of diverse slice types arrive and leave the network dynamically. Connection requests cause the dynamic partitioning of the network into a number of slices (one for each different slice type), with each slice adhering to the QoT requirement of its own slice type. The general assumption is that in the metro/core network, a slice is defined according to a set of QoS requirements (e.g., QoT requirement, bit rate, etc.) of the same type, and thus, it is not directly associated with a particular application (which is the case for fronthaul networks). Since in this work we consider only the QoT requirement, a *slice type* is formed only by the connections requesting the same BER. The aim is to find QoT models that are capable of accurately distinguishing between the diverse BER requirements. These models can then be integrated into the slice/connections provisioning phase for ensuring that each slice will meet its BER (slice type) requirement before it is actually (re)configured.

In this work, the QoT model/s estimation procedure is based on the general multi-slicing control and management framework shown in Fig. 2, consisting of a central controller (orchestrator) and several distributed controllers (e.g., fog/edge-computing-based controllers). Each controller, has its own monitoring and management capabilities, and it can store and process control and management information. Hence, under this framework, the model/s can be computed in either a centralized or distributed manner. In the first case, the central controller stores and processes global network information, whereas in the second case, each local controller stores and processes information that is relevant only to the connections intended for the slice type that each distributed controller manages (i.e., each distributed controller manages the connections of a different slice type/BER requirement). For both centralized and distributed frameworks, an ML application runs on top of a database from which it extracts and analyzes the stored data (i.e., BER data obtained via optical performance monitoring (OPM)), by means of model training. In the centralized framework, a single QoT model is trained that centrally controls/manages the network, whereas in the distributed framework, a set of QoT models is trained locally and independently, with each QoT model being responsible for handling the connection intended for its own slice type.

After the training procedure, the QoT model/s can be used during the slice provisioning phase for ensuring connection feasibility (Fig. 2). Since slice provisioning takes place in the central controller (available network resources can be more efficiently managed in a centralized manner), the distributed QoT models need to be communicated to the central controller. Note, however, that only the model parameters need to be sent to the central controller and not the entire training datasets, given that the central controller is aware of the ML method/s applied to each slice type (e.g., NN topology and activations) and that the ML method/s applied do not require the training dataset (or a subset of the training dataset) during inference (e.g., NNs do not require the training dataset, k-nearest neighbors require the entire training dataset, support vector machines require a subset of the training dataset, etc.). The QoT models (distributed or centralized) are then integrated into the multi-slice QoT-aware RSA algorithm responsible for the slice provisioning phase.

In general, the QoT model/s can be retrained offline according to the most recent dataset, for capturing the network changes (i.e., network degradation), and subsequently update the multi-slice QoT-aware RSA algorithm. Retraining can be performed either periodically or after the controller observes that the QoT model/s have drifted from a predefined acceptable number/percentage of misclassifications (i.e., a mechanism exists that triggers model retraining). To avoid the misclassification side effects, a safety margin could be used in conjunction with the QoT model (e.g., training the models according to higher BER requirements than the acceptable). The development of a retraining triggering mechanism is out of the scope of this work but constitutes an interesting research direction.

Regarding the practical implementation issues of the aforementioned frameworks, it is important to note that the existing single-slice approach of the optical network segment supports a forward error correction (FEC) scheme that tolerates a single BER requirement (the lowest acceptable among all the services). However, the support of an efficient end-to-end network slicing implementation requires the appropriate transformation of the FEC schemes (e.g., applying a FEC scheme that is designed for tolerating the diverse BER requirements). As the optical network slicing concept is currently in its infancy, such practical feasibility issues remain open research problems.

4. PROBLEM STATEMENT

We assume an elastic optical network (EON) in which connections of diverse BER requirements arrive and leave the network dynamically. The network is therefore dynamically partitioned into a number of slices that are formed according to the diverse slice types of the arriving connections. Each connection request is defined by the set $C_n^k = \{s, d, B^k\}$, where s is the source node, d is the destination node, and B^k is the BER requirement of connection n. Furthermore, $k = 1, \ldots, K$, which means that connections of K diverse BER requirements may request admission or equivalently that the network may be temporarily partitioned into up to K slice types. On this basis, the slices are dynamically (re)configured according to the arriving connections intended for each different slice type.

In an EON, one possible way for dynamically (re)configuring the diverse slice types is by solving the QoT-aware RSA problem for each arriving connection request. This, however, requires that predefined QoT estimation model/s are integrated into the RSA procedure to ensure that each computed slice type will meet its QoT requirement prior to its (re)configuration. In this work, for finding the QoT estimation model/s we apply multilayer perceptrons (MLPs), aiming to find accurate QoT models, according to the BER requirements of each slice type k. Centralized and distributed ML frameworks are examined. The trained ML-based QoT models are then integrated into the multi-slice QoT-aware RSA procedure.

5. ML-BASED QoT ESTIMATION

To find the QoT models, several ML methods can be applied (e.g., support vector machines, logistic regression, NNs, etc.). In this work, we have opted for an MLP that is a class of feedforward artificial NNs [37] with low memory and computational requirements. In general, it was shown that NNs achieve higher accuracy than other ML techniques used for finding QoT models [24], and they were also demonstrated experimentally [38,39] achieving an overall high accuracy. Comparing different ML methods is out of the scope of this work, as this work focuses mainly on comparing the advantages and limitations of the centralized and distributed frameworks proposed. Such a study, however, constitutes an interesting future direction for providing comparative results of the possible ML methods that can be applied. Note that in general, despite the ML method applied, multiclass classification problems (centralized framework) are harder to solve than their binary decompositions [33–35] (distributed framework). Hence, even though different ML methods are expected to perform differently regarding their achievable accuracy and RAM and CPU requirements, the centralized approach will still be harder to solve (with higher CPU and RAM requirements) than its distributed decompositions.

On this basis, we first describe the mathematical formulation of the MLP applied in this work. Then, we proceed to describe how our QoT estimation problem is formulated as a multiclass classifier (centralized framework) and as a set of binary classifiers (distributed framework). The following description concerns an MLP with one hidden layer. Information regarding the extension of an MLP to more hidden layers can be found in [37].

A. MLP Classification

A generic graph representation of an MLP is given in Fig. 3. The MLP in Fig. 3 consists of the input layer, one hidden layer, and the output layer. Given feature (input) vectors of the form $\mathbf{x} = \{x_j\}_{j=1}^{M}$, the input layer consists of M inputs. Given ground truth vectors of the form $\mathbf{y} = \{y_k\}_{k=1}^{K}$, the output layer consists of K outputs. The hidden layer, h, consists of M' activation units with their outputs denoted as $\{h_i\}_{i=1}^{M'}$ (their optimal number depends on the problem setting and can be found in general with trials). Given a set of training examples $(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_N, \mathbf{y}_N)$, where $\mathbf{x}_n$ is the feature vector of a sample n, and $\mathbf{y}_n \in \{0, 1\}$ is the ground truth vector of a sample n, an MLP learns the function $f(z_k)$, for each output of the MLP.

Briefly, $f(z_k)$ is the activation function applied to the kth output of the MLP:

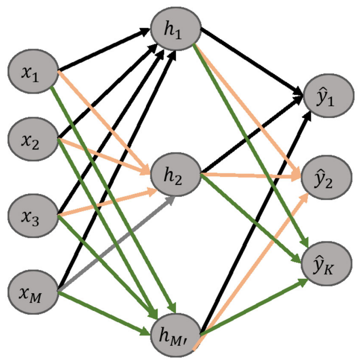

Fig. 3. Graph representation of an MLP.

$$z_k = \sum_{i=1}^{M'} w_{ik}^{(h)} h_i + b, \tag{1}$$

where b is the bias of the hidden layer, and $w_{ik}^{(h)}$ are the trainable parameters between the outputs of the hidden layer h and the kth output of the MLP [37]. Specifically, for an MLP with one hidden layer, h, the h_i outputs are calculated as follows [37] :

$$h_i = g\left(\sum_{j=1}^{M} w_{ji} x_j + b'\right), \quad \forall\, i = 1, 2, \ldots, M', \tag{2}$$

where w_{ji} are the trainable parameters between the x_j inputs and the ith hidden unit, b' is the bias, and $g()$ is the activation function applied to the hidden units. In this work, we opted for the rectified linear unit (ReLU) activation function. Information on possible activation functions and comparisons can be found in [40]. According to [40], ReLU is currently the most successful and widely used activation function and is given by

$$g(z') = (z')^+ = \max(0, z'), \tag{3}$$

where $z' = \sum_{j=1}^{M} w_j x_j + b'$.

Regarding the activation function $f()$, applied to the outputs of the MLP, for the binary classifier, where the number of classes K is equal to two (the MLP has one output and hence $f(z_k) = f(z)$), the logistic activation function is applied as follows [37]:

$$f(z) = \frac{1}{1 + e^{-z}}, \tag{4}$$

and $\hat{y} = f(z) \in \{0, 1\}$ is the score/estimate of the MLP. In general, the logistic activation function $f(z)$ maps the z output of the MLP to a binary value (zero or one), and it is typically used for binary classification.

For the multiclass classifier, where $K > 2$, the softmax activation function is applied to every k output of the MLP as follows [37]:

$$f(z_k) = \frac{e^{z_k}}{\sum_j^K e^{z_j}}, \tag{5}$$

and $\hat{y}_j = 1$ if $f(z_j) = \max\{f(z_k)\}_{k=1}^K$; otherwise $\hat{y}_j = 0$ (the output is a vector $\hat{\mathbf{y}} = \{\hat{y}\}_{j=1}^K$, and the maximum score of $\{f(z_k)\}_{k=1}^K$ indicates the class that sample $\mathbf{x}$ belongs to). Note that the softmax activation function $f(z_k)$ is typically used for multiclass classification purposes and maps the outputs z_k of the MLP to scores (probabilities). Then, the class with the highest probability is chosen for the input sample $\mathbf{x}$.

In order to train the MLP, the crossentropy loss function is used, optimized in accordance with the Adam algorithm [41]. Specifically, for multiclass classification, the categorical crossentropy is used [42]:

$$\text{CE}_{\text{loss}} = -\sum_k^K y_k \log(f(z_k)), \tag{6}$$

where $y_k \in \mathbf{y}$ is the ground truth of a sample $\mathbf{x}$, and $f(z_k)$ [Eq. (5)] is the score of the MLP for class k. For the binary classification case, the crossentropy loss function reduces to [42]

$$\text{CE}_{\text{loss}} = y \log(f(z)) - (1 - y) \log(1 - f(z)), \tag{7}$$

where y is the ground truth of a sample $\mathbf{x}$, and $f(z)$ is the score of the MLP [Eq. (4)].

B. Centralized QoT Problem Formulation

The centralized QoT estimation problem is formulated as a multiclass classifier for an EON. In general, a multiclass classification problem can be briefly described as follows: given a dataset $D = (\mathbf{X}, \mathbf{Y}) = \{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^N$, where $\mathbf{x}_n$ is an input vector describing the features of pattern n, and $\mathbf{y}_n$ is its ground truth vector describing the class that pattern n belongs to, find a model that accurately predicts the class that any unseen pattern n' belongs to. For our QoT estimation problem, $\mathbf{x}_n$ is a vector describing the features of lightpath n, and $\mathbf{y}_n$ is the vector describing the BER ground truth of lightpath n. Thus, the objective is to find, from D, a QoT model $f = \hat{\mathbf{y}}$ that accurately estimates the QoT class of any unseen lightpath n'.

Feature vector $\mathbf{x}_n = \{x_j\}_{j=1}^M$ describes lightpath n as follows:

- x_1: is the entire length (in km) of lightpath n.
- x_2: is the maximum link length (in km) of lightpath n.
- x_3: is the central frequency allocated to lightpath n.
- x_4: is the number of slots allocated to lightpath n.
- x_5: encodes the modulation format for lightpath n. Specifically, $x_5 = 1$ for binary phase shift keying (BPSK), $x_5 = 2$ for quadrature phase shift keying (QPSK), $x_5 = 3$ for 8-quadrature amplitude modulation (8-QAM) and $x_5 = 4$ for 16-QAM.
- x_6: denotes the number of erbium doped fiber amplifiers (EDFAs) along lightpath n.
- x_7: is the number of links along lightpath n.

Vector $\mathbf{y}_n = \{y_j\}_{j=1}^{K+1} \in \{0, 1\}$ declares the QoT class of lightpath n. Specifically, $y_j = 1$ if lightpath n belongs to class j, and $y_j = 0$, otherwise. Regarding the QoT classes, these are defined according to a set of all possible BER requirements. Specifically, given $\mathbf{B} = \{B^k\}_{k=1}^K$, where $\mathbf{B}$ is the set of all possible BER requirements, K is the number of all possible slice types, and $B^k < B^{k+1}$, then $K + 1$ QoT classes can be defined

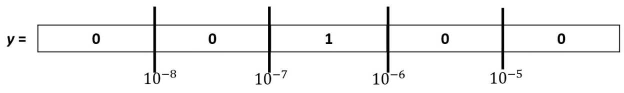

Fig. 4. Example: encoding the centralized BER ground truth for a connection with monitored BER $= 5 \times 10^{-7}$.

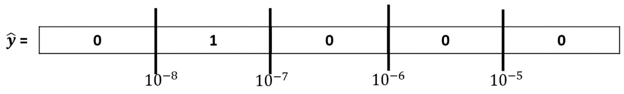

Fig. 5. Example: centralized model output declaring a feasible QoT for a connection with BER requirement equal to 10^{-6}.

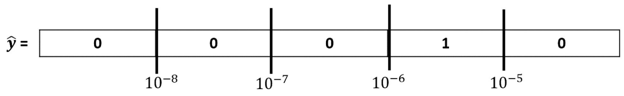

Fig. 6. Example: centralized model output declaring an infeasible QoT for a connection with BER requirement equal to 10^{-6}.

according to the ground truth BER_n of each lightpath n as follows:

- Class 1: if $\mathrm{BER}_n < B^1$, then $y_1 = 1$.
- Class v: if $B^{v-1} \leq \mathrm{BER}_n < B^v$, then $y_v = 1\, \forall 1 < v \leq K$.
- Class $K+1$: if $\mathrm{BER}_n > B^K$, then $y_{K+1} = 1$.

As an example, consider that $\mathbf{B} = \{B^1, B^2, B^3, B^4\} = \{10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}\}$, which results in five possible classes. The five classes are illustrated in Fig. 4 through the ground truth vector $\mathbf{y}$. Given a lightpath n with a ground truth (monitored) $\mathrm{BER}_n = 5 \times 10^{-7}$, its $\mathbf{y}$ vector is encoded as shown in Fig. 4. Specifically, since $10^{-7} < \mathrm{BER}_n < 10^{-6}$, a one is placed for class 3 of the $\mathbf{y}$ vector, and zeros are placed for the rest of the classes.

In this work, the QoT model $f = \hat{\mathbf{y}}$ is obtained based on the MLP described in Section 5.A. An unseen vector $\mathbf{x}_{n'}$ (which has not been used during the training procedure) is the input to the trained QoT model, and output vector $\mathbf{y}_{n'}$ is subsequently returned. If $\hat{y}_v = 1$ and the BER requirement of lightpath n' is B^j, then lightpath n' is feasible (i.e., its $\mathrm{BER}_{n'}$ is lower than the B^j requirement) if $v < j$; otherwise it is infeasible.

As an example consider Figs. 5 and 6. If we assume that an arriving connection n' has a BER requirement equal to $B^3 = 10^{-6}$ and the trained model returns $\hat{y}_2 = 1$ (Fig. 5), then the computed lightpath for connection n' is considered as feasible, as the estimated BER is lower than the connection's BER requirement. If, however, the trained model returns $\hat{y}_4 = 1$ (Fig. 6), then the computed lightpath for connection n' is considered as infeasible, as the estimated BER is higher than the connection's BER requirement.

C. Distributed QoT Problem Formulation

The distributed QoT estimation problem is formulated according to a set of binary classifiers for an EON. Each binary classifier is trained according to a dataset D^k, where D^k is the dataset stored in the local controller of slice type k. If K is the total number of slice types/BER requirements, then $\mathbf{D} = \{D^k\}_{k=1}^K$ is the set consisting of all the distributed datasets. The objective is to find from $\mathbf{D}$ a set of QoT models $\mathbf{f} = \{f^k\}_{k=1}^K$, where f^k is the QoT model of slice type k.

Specifically, $D^k = (\mathbf{x}^k, y^k) = \{\mathbf{x}_n^k, y_n^k\}_{n=1}^{N'}$, where $\mathbf{x}_n^k$ is the features' vector of lightpath n, y_n^k encodes the BER ground truth of lightpath n, and N' is the number of patterns in dataset D^k. Note that D^k describes a set of lightpaths established/intended only for slice type k. The $\mathbf{x}_n^k$ vector is as previously defined in Section 5.B. The ground truth y_n^k is encoded in a binary manner. Specifically, $y_n^k = 0$ when the BER of lightpath n is above its BER requirement B^k (class 1—infeasible class), and $y_n^k = 1$ otherwise (class 2—feasible class).

Each distributed QoT model, $f^k \in \mathbf{f}$, is trained locally and independently from the other QoT models according to an MLP. Once the model is trained, the $\mathbf{f}$ model parameters are communicated to the central controller and integrated into the QoT-aware connection provisioning procedure that is executed centrally. During inference, the f^k model takes as input an unseen feature vector $\mathbf{x}_{n'}^k$ and returns $\hat{y}_{n'}^k$. Lightpath n' is feasible if $\hat{y}_{n'}^k = 1$; otherwise it is infeasible.

6. QoT-AWARE SLICE PROVISIONING

To establish a connection of any slice type k, the QoT-aware RSA problem must be solved. To solve the QoT-aware RSA problem, we developed simple multi-slice QoT-aware RSA heuristics that integrate both the centralized and distributed ML-based QoT models. As a benchmark, we have also developed the single-slice QoT-aware RSA, which does not distinguish between the diverse QoT requirements/slice types and is based on a single ML-based QoT model that utilizes the lowest BER requirement among the diverse BER requirements. All heuristics developed follow the same RSA procedure and differ only in the way the QoT of each connection request is estimated and considered. In particular, all heuristics for each arriving connection C_n^k solve the routing (R) problem utilizing Dijkstra's algorithm [43], and the spectrum allocation (SA) problem utilizing the first-fit approach. According to the first-fit algorithm, the heuristics allocate to each shortest path the first feasible spectrum slots, such that the spectrum continuity, contiguity, and no-frequency overlap constraints are met. The QoT for each computed lightpath is then estimated. If the estimated QoT is sufficient, then the lightpath is established in the network; otherwise it is blocked.

A. Multi-Slice QoT-Aware RSA

1. Centralized ML-Based QoT Models

Given f, where f is a multiclass QoT estimation classifier trained according to the set of all possible BER requirements $\mathbf{B}$, the multi-slice QoT-aware RSA objective is to find for each C_n^k connection a route and an SA that meets the SA constraints (i.e., spectrum continuity, contiguity, and no-frequency overlap), along with its corresponding QoT requirement as defined by the f model. As a reminder, $C_n^k = \{s, d, B^k\}$, where s–d is the pair of source–destination nodes, and B^k is the BER

Algorithm 1. Multi-Slice QoT-Aware RSA Heuristic Utilizing the Centralized ML-Based QoT Model

```
Input: QoT model $f$, $K$ diverse BER requirements (slice types), connection arrival rate $\lambda$ and departure rate $\mu$
Output: Number of blocked connections
 1: for each arriving request do
 2:     Calculate the shortest path for connection $C_n^k$
 3:     For the shortest path of $C_n^k$, choose the first-fit SA that meets
        the SA constraints (i.e., spectrum continuity, contiguity, and
        no-frequency overlap).
 4:     if lightpath $n$ is not feasible then
 5:         Set $\hat{\mathbf{y}} = 0$
 6:     else
 7:         Randomly set an element $\hat{y}_{v'} \in \mathbf{y}$ to 1 and the rest to 0 so
        that $\sum \hat{\mathbf{y}} = 1$
 8:     if $\sum \hat{\mathbf{y}} = 1$ then
 9:         Extract the feature vector $\mathbf{x}_n$ from lightpath $n$
10:         Set $\hat{\mathbf{y}} = 0$
11:         Estimate $\hat{\mathbf{y}} = f$
12:     if $\hat{y}_v = 1$ AND $v < k$ then
13:     $C_n^k$ is admitted into the network
14:     else
15:         $C_n^k$ is blocked
    return The number of blocked connections
```

requirement of connection n. Model f is trained in a centralized manner as a multiclass classifier (Section 5.B). Hence, f returns $\hat{y}_v = 1$ and $\hat{y}_j = 0 \, \forall j \neq v$ if the BER of connection n is below the predetermined threshold B^k (i.e., $v < k$). In that case, the connection is admitted into the network; otherwise, if $\hat{y}_v = 1$ and $v \geq k$, the connection is blocked. The description of the multi-slice QoT-aware RSA heuristic utilizing the centralized ML-based QoT model is shown in Algorithm 1.

2. *Distributed ML-Based QoT Models*

Given $\mathbf{f} = \{f^k\}_{k=1}^K$, where f^k is a binary QoT estimation classifier trained according to B^k, the multi-slice QoT-aware RSA objective is to find for each C_n^k connection a route and an SA that meets the spectrum continuity, contiguity, and no-frequency overlap constraints, along with its corresponding QoT requirement as defined by the f^k model. For the set of QoT models, $\mathbf{f}$, each f^k model is trained locally and independently (Section 5.C). Hence, for each $f^k \in \mathbf{f}$, f^k returns $\hat{y} = 1$ if the BER of the connection is below the predetermined threshold B^k, and zero otherwise. If $\hat{y} = 1$, the connection is admitted into the network; otherwise it is blocked. The description of the multi-slice QoT-aware RSA heuristic utilizing the distributed ML-based QoT models is shown in Algorithm 2.

B. Single-Slice QoT-Aware RSA

Given $\mathbf{f} = f^m$, where f^m is a binary QoT estimation classifier trained according to $B^m = \min\{B^k\}_{k=1}^K$, the objective of the single-slice QoT-aware RSA is to find for each C_n^k connection a route and an SA that meets the spectrum continuity, contiguity, and no-frequency overlap constraints, along with its

Algorithm 2. Multi-Slice QoT-Aware RSA Heuristic Utilizing the Distributed ML-Based QoT Models

```
Input: Set of QoT models $\mathbf{f} = \{f^k\}_{k=1}^K$, $K$ diverse BER requirements (slice types), connection arrival rate $\lambda$, and departure rate $\mu$
Output: Number of blocked connections
 1: for each arriving request do
 2:     Calculate the shortest path for connection $C_n^k$
 3:     For the shortest path of $C_n^k$, choose the first-fit SA that meets
        the SA constraints (i.e., spectrum continuity, contiguity, and
        no-frequency overlap).
 4:     if lightpath $n$ is not feasible then
 5:         $\hat{y} = 0$
 6:     else
 7:         $\hat{y} = 1$
 8:     if $\hat{y} = 1$ then
 9:         Extract the feature vector $\mathbf{x}_n$ from lightpath $n$
10:         Estimate $\hat{y} = f^k$
11:     if $\hat{y} = 1$ then
12:         $C_n^k$ is admitted into the network
13:     else
14:         $C_n^k$ is blocked
    return The number of blocked connections
```

corresponding QoT requirement as defined by the f^m model. For the QoT model, f^m is trained as a binary classifier according to the MLP described in Section 5.C. Hence, $\hat{y} = 1$ if the BER of the connection is below the predetermined threshold B^m, and zero otherwise. If $\hat{y} = 1$, the connection is admitted into the network; otherwise it is blocked. The description of the single-slice QoT-aware RSA is not given; however, it is similar to the one shown in Algorithm 2 with the difference that only one binary QoT classifier is used for the QoT feasibility decisions.

7. DATASET GENERATION AND MODEL TRAINING AND EVALUATION

A. Dataset Generation

For the dataset generation procedure, we used the Telefonica network topology (shown in Fig. 7). An EON is assumed, which is implemented using bandwidth variable transponders that can operate with BPSK, QPSK, 8-QAM, and 16-QAM modulation formats. The channel spacing for this network is set to 25 GHz, the guard band is set to 25 GHz, and the baud rate is set to 16 Gbaud for a total of 160 frequency slots for each network link. Twenty-five thousand ($N = 25{,}000$) connection requests were generated in a dynamic network following a Poisson process with exponentially distributed holding times for several network loads varying between 50 to 500 Erlangs. Specifically, we have performed 10 simulation runs, and for each simulation run, 2500 connection requests were generated. The datasets extracted from all the runs were merged into a single dataset subsequently used for training/validating the ML-based QoT models.

Each connection request $C_n^k = \{s, d, B^k\}$ was generated as follows: the *s–d* (source–destination) pairs were generated by randomly sampling from the set of network nodes; the B^k (BER) requirement of the nth connection request was

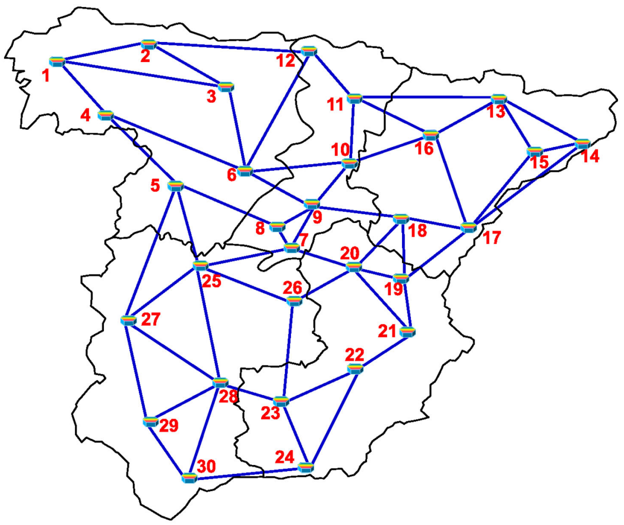

Fig. 7. Telefonica network topology.

generated by randomly sampling from a set of possible BER requirements $\mathbf{B} = \{B^k\}_{k=1}^K$; a bit rate was randomly generated for each connection request by uniformly sampling within the interval [10,200] Gbps.

For the connection provisioning phase, a conventional RSA algorithm was applied. Dijkstra's shortest path algorithm [43] was used for the R sub-problem, and the first-fit technique was utilized for the SA sub-problem, while at the same time ensuring that all three SA constraints are met. For each lightpath provisioned, the Q-tool (described in [28]) was used for estimating the ground truth values (i.e., BER_n). In practical implementations, the BER ground truth values can be extracted trough OPM at the receivers [11]. Furthermore, probing lightpaths or alien wavelengths [11,38] can also be utilized to supplement the dataset information, especially for the connections with insufficient QoT. Moreover, transfer learning [44] or active learning [45] techniques can be applied for reducing the number of probes required for QoT model training.

1. Centralized Dataset

The multiclass dataset $D = \{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^N$ was created by extracting from each computed lightpath n its $\mathbf{x}_n$ and $\mathbf{y}_n$ vectors. Section 5.B analytically describes both the lightpaths' features (specified in vector $\mathbf{x}_n$), as well as the encoding procedure of the ground truth BER_n in vector $\mathbf{y}_n$.

2. Distributed Datasets

The distributed datasets $\mathbf{D} = \{D^k\}_{k=1}^K$ were created by partitioning the multiclass dataset D into K binary datasets. Specifically, each distributed dataset $D^k = \{\mathbf{x}_n^k, y_n^k\}_{n=1}^{N'}$ was generated by selecting from D only the patterns (connections) with a BER requirement that is equal to B^k. Hence, feature vectors $\mathbf{x}_n^k$ can be found in D, while y_n^k takes the value one if the ground truth BER_n of lightpath n is below B^k; otherwise it takes the value zero. Note that in our simulations (for simplifying the dataset generation procedure of the diverse datasets) the sum of patterns in the distributed datasets $\mathbf{D}$ may be less than the number of patterns in the multiclass dataset D depending on the B^k requirements considered for the creation of the D^k datasets. Specifically, if the number of diverse B^k requirements considered in the distributed case is less than the number of BER requirements considered for generating dataset D, then the number of patterns in $\mathbf{D}$ will be less than the number of patterns in D.

B. Model Training

Both centralized and distributed QoT models were trained according to an MLP with one hidden layer and 20 hidden units. The centralized QoT models were additionally trained according to a larger MLP topology consisting of three hidden layers and 1000 units each. In practice, the centralized multiclass classification problem is in general harder to solve, and hence a larger MLP topology may be needed to achieve better classification accuracy. The MLPs used for training the centralized multiclass classifier consist of $K + 1$ outputs (equal to the total number of classes in dataset D), while the MLP used for training the distributed binary classifiers has one output. The mathematical formulation of the MLP applied for both the multiclass classifier and the binary classifiers, consisting of one hidden layer, is analytically described in Section 5. Information regarding MLPs with more hidden layers can be found in [37].

Note that different MLP training trials were performed examining different numbers of units and layers. We have chosen to show the results of a few MLP configurations achieving sufficient classification accuracy for both the centralized and distributed frameworks, and also highlighting the trade-offs between the multiclass classification approach and its alternative binary decompositions. In general, by increasing the number of hidden layers, higher accuracy can be achieved, but complexity of the MLP and training time are increased manyfold. Furthermore, unnecessary increments in the neurons and/or layers lead to overfitting [46].

The Adam [41] algorithm was used for optimizing the loss function applied to the output layer, for a total of 30 epochs and a batch size equal to 50. Dropout regularization [47] was applied to Adam, and the dropout rate was set to 0.5 (which is close to optimal for a wide range of networks and tasks according to [47]). In general, dropout is a regularization method for preventing units from co-adapting too much by randomly dropping units (along with their connections) from the MLP during training (for each batch). The resulting network (i.e., during testing) is used without dropout. Specifically, during testing, all the units and connections are in place (i.e., the trainable parameters are scaled over all the batches and used for testing).

The learning rate was set to 0.01, and the size of the validation dataset was set to 1/3 of the total number of patterns in dataset D. Three-fold cross validation was performed, and the model's accuracy was averaged over all the folds. It is important to note that dataset D was standardized (scaled) so as the data features to be standard normally distributed before training and testing. Standardization of a dataset is a common requirement for many ML estimators, since they might otherwise behave badly.

C. Model Evaluation

After the training procedure, both the centralized and distributed QoT models were evaluated and compared according to their accuracy, accuracy per class, CPU time, and RAM usage. To train and test the QoT models, a PC with an i7-3930K CPU, a 6 GB GTX 1660Ti GPU, and 24 GB RAM was used.

1. *Centralized Model*

The centralized framework was examined for three cases that differ according to the sets of possible BER requirements. Specifically, in the first case, the set of BER requirements assumed is $\mathbf{B}=\{10^{-8}, 10^{-6}, 10^{-4}\}$ (i.e., $K=3$ slice types), in the second case $\mathbf{B}=\{10^{-8}, 10^{-7}, 10^{-6}, 10^{-4}\}$ (i.e., $K=4$ slice types), and in the third case $\mathbf{B}=\{10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}\}$ (i.e., $K=5$ slice types). Note that these sets of BER requirements were chosen so that the BER ground truth for each connection request can be estimated utilizing the Q-tool [28].

Therefore, three QoT models were trained that differ based on their number of classes. Specifically, as K slice types (i.e., BER requirements) lead to $K+1$ classes, three classifiers were trained and tested consisting of four, five, and six classes. Table 1 provides information regarding the total number of patterns (training plus testing patterns) that belong to each class. Note that 16,667 patterns were used for model training and 8333 for model inference/testing. Tables 2 and 3 present results on the models' accuracy, accuracy per class, CPU, and RAM usage for the two MLP topologies examined, i.e., the MLP with one hidden layer with 20 units and the MLP with three hidden layers with 1000 units each, respectively.

Figures 8 and 9 illustrate how the model's accuracy and loss evolve (converge) with the number of epochs, respectively. These figures correspond, indicatively, to the $K=5$ case for the MLP with one hidden layer, while similar convergence behavior has also been observed for the $K=3$ and $K=4$ cases and the MLP with three hidden layers, i.e., by increasing the number of epochs, the model's accuracy/loss does not improve. Note that according to Fig. 8, the model seems to converge well before it reaches 30 epochs. However, Fig. 9 shows that the loss function keeps decreasing up to 30 epochs, an indicator that the model achieves better generalization accuracy (avoids overfitting). Specifically, the method avoids overfitting, as the test loss function keeps decreasing. Note that in Fig. 8, the test accuracy is slightly higher than the training accuracy,

Table 1. Number of Patterns per Class for the Centralized QoT Model

	4 Classes ($K=3$)	5 Classes ($K=4$)	6 Classes ($K=5$)
Class 1	10,083	10,083	10,083
Class 2	8895	4323	4323
Class 3	5199	4572	4572
Class 4	823	5199	4755
Class 5	-	823	444
Class 6	-	-	823

Table 2. Accuracy Results for the Centralized QoT Model (1 hidden layer with 20 units)

	4 Classes ($K=3$)	5 Classes ($K=4$)	6 Classes ($K=5$)
Model acc. (%)	94	90	90
Class 1 acc. (%)	96	96	96
Class 2 acc. (%)	92	82	82
Class 3 acc. (%)	93	80	79
Class 4 acc. (%)	100	93	92
Class 5 acc. (%)	-	100	83
Class 6 acc. (%)	-	-	100
CPU usage (s)	68	68	71
RAM usage (MB)	283	283	283

Table 3. Accuracy Results for the Centralized QoT Model (3 hidden layers with 1000 units each)

	4 Classes ($K=3$)	5 Classes ($K=4$)	6 Classes ($K=5$)
Model acc. (%)	95	92	91
Class 1 acc. (%)	96	97	96
Class 2 acc. (%)	93	83	86
Class 3 acc. (%)	94	82	82
Class 4 acc. (%)	100	93	91
Class 5 acc. (%)	-	100	86
Class 6 acc. (%)	-	-	100
CPU usage (s)	1026	1096	1094
RAM usage (MB)	419	428	429

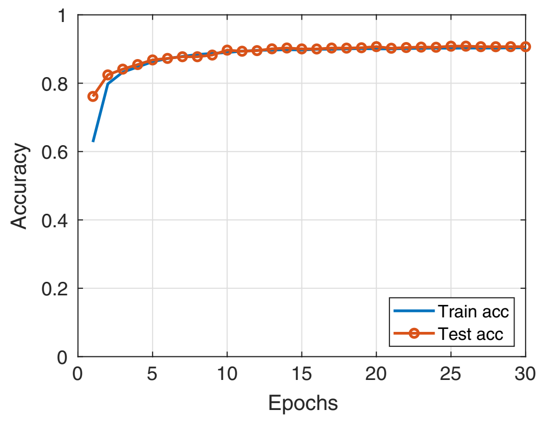

Fig. 8. Centralized model: accuracy versus the number of epochs for $K=5$ (MLP with one hidden layer).

and in Fig. 9, the test loss is slightly lower than the training loss. A common reason behind this outcome is the dropout regularization [47] applied during Adam optimization [41]. In particular, the dropped MLP units and connections during training may lead to a lower training accuracy (higher training loss) than the testing accuracy (testing loss), since during testing all the units and connections are considered.

According to Tables 2 and 3, the overall model accuracy is sufficiently high for all three cases examined, while the MLP with three hidden layers slightly improves both the overall

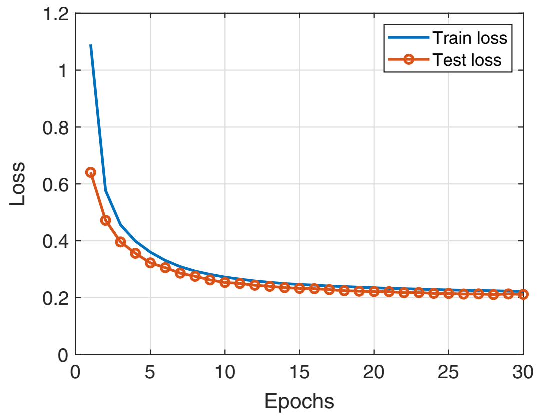

Fig. 9. Centralized model: loss versus the number of epochs for $K = 5$ (MLP with one hidden layer).

classification performance and the per class classification performance, an indicator that the increment in both hidden layers and units favors performance accuracy, at the expense of CPU and RAM usage. However, for both MLP topologies examined (a small and a large MLP topology), as the number of classes increases, the model's overall accuracy slightly drops. Further, the model's accuracy per class is also high (for the majority of the classes). However, as the number of classes increases, this metric decreases significantly. Indicatively, the model's accuracy per class drops to approximately 82% for the six classes (Table 3), below an accuracy level that is considered acceptable. This is to be expected, considering that, as the number of classes increases, the problem of distinguishing between the various classes becomes harder, i.e., the success rate of each class drops as the number of classes increases. Moreover, as the BER requirements increase, the training dataset is partitioned into more classes with reduced patterns (Table 1), with sizes that may not be large enough for achieving a high accuracy per class. Note that the F1-score (representing the harmonic mean of precision and recall) [48] was also evaluated for each one of the classes to check if the imbalanced classes (Table 1) affect the classification performance. As the F1-scores were similar to the accuracy results (slightly different but with the same classification performance, especially as the number of classes increases), we have chosen to analytically present only the results regarding the accuracy, a metric that is widely accepted and more easily interpreted.

Recently, it has been shown that deep NN training can always achieve the highest inference accuracy, so long as sufficiently large amounts of data samples are fed for training, emphasizing the importance of sufficient data sample acquisition [49,50]. On this basis, for our QoT estimation problem, although the accuracy per class may increase if the number of diverse patterns increases (especially within each class that underperforms), such a procedure inevitably increases the RAM and CPU usage of the MLP training [49,50], as well as the complexity of the dataset generation process.

Specifically, in practice, extra probing lightpaths will need to be "intelligently" identified and established, aiming to add valuable information into the dataset (i.e., random probes will only increase the network load without ensuring that any additional/valuable information will be extracted). The development of such a mechanism is out of the scope of this work, but it constitutes an interesting topic for future work. In this work, we opted to utilize the same dataset for both the centralized and distributed QoT model training frameworks so as to fairly compare these frameworks for every relevant metric examined. By doing so, we managed to showcase the drawbacks and advantages of each approach.

Regarding both the CPU and RAM usage, as shown in Tables 2 and 3, these do not significantly change with the number of classes. Specifically, the CPU and RAM usage in Table 2 is the same for all three cases. This is reasonable, as these depend mainly on the number of training patterns, MLP configuration, batch size, and number of epochs, which are the same for all three models presented in Table 2. However, by comparing Tables 2 and 3, clearly both the CPU and RAM usage increase for the larger MLP. Hence, the higher accuracy achieved with the MLP with a larger number of hidden layers and units is achieved at the expense of an overall higher CPU and RAM usage.

2. *Distributed Models*

For the distributed framework, the two cases of interest that were examined were the cases for which the centralized framework underperformed. Specifically, it was examined for the case where $\mathbf{B} = \{10^{-8}, 10^{-7}, 10^{-6}, 10^{-4}\}$ ($K = 4$) and for the case where $\mathbf{B} = \{10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}\}$ ($K = 5$). Thus, for these cases, four and five binary classifiers were trained, respectively. Note that only the MLP topology consisting of one hidden layer and 20 units was examined for the distributed framework, as the accuracy achieved by this topology was sufficiently high. Moreover, the multiclass classificiaton problem is in general harder to solve than its alternative binary decompositions, and hence it is reasonable that a smaller MLP topology will be sufficient for the binary classifiers.

Tables 4 and 6 provide dataset information regarding the total number of patterns (training plus testing patterns) per binary classifier, and the number of patterns per class. Tables 5 and 7 summarize the results for the $K = 4$ and $K = 5$ binary classifiers, respectively. These results include the overall accuracy, the accuracy per class, as well as the CPU and RAM usage. Note that the four classifiers out of the five trained for the $K = 5$ case are the same as those trained for the $K = 4$ case. This is because the $\mathbf{B}$ set assumed for the $K = 5$ case includes all the BER requirements assumed for the $K = 4$ case plus one BER requirement that is placed between the 10^{-6}–10^{-4} thresholds. Due to the imbalanced classes (Tables 4 and 6), the F1-score was also evaluated for each distributed case, exhibiting a classification performance similar to the accuracy results. Hence, the F1-scores are not analytically presented in this work.

Figures 10 and 11 illustrate how the model's accuracy and loss evolve (converge) with the number of epochs, respectively. These figures correspond, indicatively, to the $K = 5$, $k = 3$ case (Table 7), while similar convergence behavior has been observed for all other classifiers examined, i.e., by increasing the number of epochs, the model's accuracy/loss does not improve. Similar to the centralized framework, even though

Table 4. Dataset Information for the Distributed QoT Models ($K = 4$)

Slice Type (k)	D^k Patterns	Class 1 Patterns	Class 2 Patterns
1	3660	1515	2145
2	4137	1344	2793
3	1869	441	1428
4	9546	831	8715

Table 5. Results for the Distributed QoT Models ($K = 4$)

Slice Type (k)	Model Acc. (%)	Class 1 Acc. (%)	Class 2 Acc. (%)	CPU (s)	RAM (MB)
1	98	96	99	9	275
2	98	97	99	10	281
3	98	96	99	7	277
4	100	100	100	17	280

Table 6. Dataset Information for the Distributed QoT Models ($K = 5$)

Slice Type (k)	D^k Patterns	Class 1 Patterns	Class 2 Patterns
1	3660	1515	2145
2	4137	1344	2793
3	1869	441	1428
4	4767	315	4452
5	9546	831	8715

Table 7. Results for the Distributed QoT Models ($K = 5$)

Slice Type (k)	Model Acc. (%)	Class 1 Acc. (%)	Class 2 Acc. (%)	CPU (s)	RAM (MB)
1	98	96	99	9	275
2	98	97	99	10	281
3	98	96	99	7	277
4	99	91	99	10	277
5	100	100	100	17	280

the model seems to converge well before 30 epochs (Fig. 10), Fig. 11 suggests that the model achieves better generalization accuracy (avoids overfitting) as the number of epochs increases, i.e., the test loss function keeps decreasing. Furthermore, according to Figs. 10 and 11, the training accuracy (training loss) is slightly lower (higher) than the test accuracy (test loss). As previously mentioned, the most common reason behind this effect is the dropout regularization applied to the Adam optimization algorithm.

These performance results clearly demonstrate that an overall high accuracy (exceeding 98%) is achieved for all QoT models. Furthermore, unlike the centralized case, all the models achieve an overall high accuracy per class (i.e., for the infeasible and feasible classes 1 and 2, respectively). Further, the CPU time results obtained for all models are between 7 and 17 s. This is reasonable, as the CPU time depends, apart from the MLP configuration and batch size (which are the same for

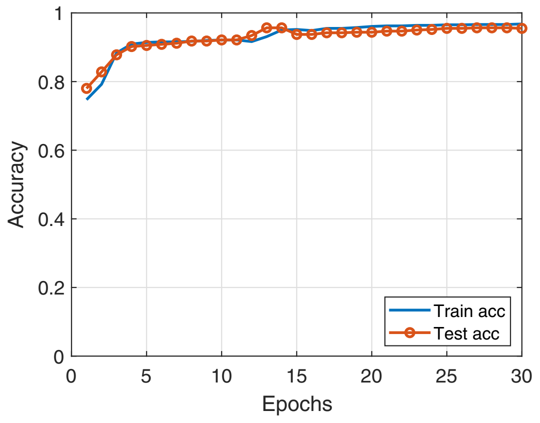

Fig. 10. Distributed models: accuracy versus the number of epochs for $K = 5$, $k = 3$.

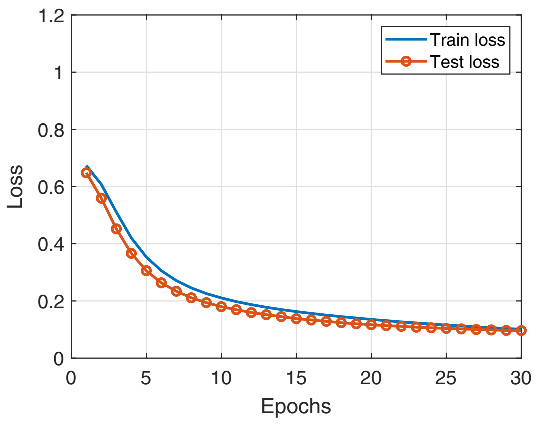

Fig. 11. Distributed models: loss versus the number of epochs for $K = 5$, $k = 3$.

each distributed classifier), on the number of patterns used for training and validating each binary dataset as well. According to Tables 4 and 6, the dataset size for each binary classifier k varies, as each binary classifier k is trained only according to the patterns/lightpaths that have a BER requirement equal to B_k. Tables 5 and 7 clearly show that the CPU time increases as the number of patterns increases.

The RAM usage (Tables 5 and 7) is roughly the same for all binary classifiers examined, as it is affected mainly by the MLP topology (number of hidden layers and units) and negligibly affected by the training datasets (which vary but are in general small in size). As the MLP topology used for all binary classifiers is the same, the RAM usage does not vary significantly.

D. Comparing the Centralized and Distributed Models

It is clearly demonstrated that the distributed framework greatly outperforms the centralized framework in terms of both the models' overall accuracy and the models' accuracy per class. Importantly, the results indicate that the accuracy of the centralized framework decreases when the number of slice types increases, while the accuracy of the distributed framework is independent of the number of slice types. This is to be expected, if we consider that the distributed QoT models are always binary classifiers, and hence they always have to distinguish between only two classes. On the other hand, the centralized QoT model is a multiclass classifier that depends on the number of slice types. Thus, in this case, with an increasing number of slice types, the difficulty level for the classifier to distinguish between the classes (BER requirements) increases significantly. While it is possible that by "intelligently" increasing the number (type) of patterns in the centralized dataset sufficient accuracy may be achieved, the results indicate that the distributed approach requires both less information for achieving an overall sufficient accuracy and fewer trainable parameters in the MLP topology.

To further compare the distributed and centralized framework, we have evaluated one more metric, namely, the RSA accuracy metric. This metric shows whether the misclassified patterns in the centralized and distributed frameworks lead to incorrect RSA decisions, i.e., a lightpath that is classified in the wrong class may still lead to the right RSA decision. This metric is readily available for the distributed framework, as the distributed models directly indicate whether a lightpath is feasible or not, i.e., directly indicate whether a lightpath is below (feasible) or above (infeasible) its B^k requirement. The centralized framework, however, classifies the patterns into possible BER ranges, and thus the feasibility of the lightpaths with regard to the their B^k requirement is not directly interpreted after model training and testing. Hence, this metric is evaluated to further investigate the multiclass classifier according to the QoT-aware RSA decisions.

For the distributed model, the RSA accuracy is given by just averaging the models' accuracy after model training and testing. Similarly, the RSA accuracy per class is given by just averaging the models' accuracy per class after model training and testing. Both the RSA accuracy and the RSA accuracy per class measure how the distributed models jointly perform in the network during the QoT-aware RSA decisions. For the centralized framework, the test outputs of the model (predicted values) and the test ground truth values were compared against the B^k requirement allocated to each unseen lightpath. Specifically, the test outputs of the model and their test ground truth values were mapped to a binary value indicating whether a lightpath is feasible or not, depending on the B^k of each unseen lightpath. The classes indicating a range of BER values above the B^k value were mapped to the zero value (RSA class 1), indicating an infeasible lightpath. The output classes indicating a range of BER values below the B^k value were mapped to the one value (RSA class 2), indicating a feasible lightpath.

Hence, for the centralized framework, the RSA accuracy was evaluated by comparing the mapped values of the test outputs with the mapped values of their test ground truths. The RSA accuracy is then the number of correctly classified outputs (according to their mapped values) over the number of test outputs. The RSA accuracy for class 1 (class 2) is the number of outputs correctly classified to the RSA class 1 (class 2) over the number of test outputs that truly belong to the RSA class 1 (class 2), according to their mapped ground truth values.

Table 8. RSA Accuracy Results for the Centralized QoT Models

	4 Classes ($K=3$)	5 Classes ($K=4$)	6 Classes ($K=5$)
RSA acc. (%)	98.6	98	98.5
RSA class 1 acc. (%)	96.5	96	95.7
RSA class 2 acc. (%)	99	99	99

Table 9. RSA Accuracy Results for the Distributed QoT Models

	$K=4$	$K=5$
RSA acc. (%)	98.6	98.5
RSA class 1 acc. (%)	97	96
RSA class 2 acc. (%)	99	99

Tables 8 and 9 show the RSA accuracy and the RSA accuracy per class values for all the centralized and distributed cases examined, respectively. Note that for the centralized case, we used the QoT model that was trained according to the MLP with three hidden layers (the RSA results of the one hidden layer were slightly lower), while for the distributed case, we used the QoT models that were trained according to the MLP with one hidden layer. According to the results, both the centralized and the distributed frameworks perform sufficiently high regarding the QoT-aware RSA decisions with the distributed framework performing slightly better, an indicator that although the centralized framework may fail to achieve a high accuracy for all the BER classes, the QoT-aware RSA decisions may not be affected. Note, however, that for the centralized framework, an extra step is required for validating the QoT-aware RSA decisions, while for the distributed framework, these results are readily available after model training and testing.

Furthermore, according to Tables 8 and 9, approximately 3%–4% of the infeasible (negative) patterns (for both frameworks) will be false positives, and hence will be accepted rather than blocked. On the other hand, 1% of the feasible (positive) patterns will be false negatives, and hence will be blocked rather than accepted. Overall, the blocking will be negatively affected (increased) by approximately 1% due to the false negatives, even though approximately 3%–4% of the negative patterns will be accepted rather than blocked. Nevertheless, according to the high accuracy results, the RSA performance will not be significantly affected. For handling the false positives (it is more critical to accept a pattern with insufficient BER rather than blocking a pattern with sufficient BER), a possible solution is to train the QoT models according to BER thresholds that are slightly higher than the BER requirements of each slice type. Doing so, a safety margin can be included in the trained model/s resembling the safety margin traditionally included in the Q-tool models.

Regarding the CPU usage, according to the results of Tables 2, 3, 5, and 7, it increases as the number of training patterns increases and as the MLP topology (number of units and layers) increases. In general, the CPU usage is significantly lower per distributed model compared to the centralized models, especially when compared to the larger MLP topology used for training the centralized models.

Regarding the RAM usage, as it is affected mainly by the MLP topology, it is approximately the same for all the models trained according to the MLP topology with one hidden layer. However, for the MLP with three hidden layers, used for improving the accuracy of the centralized multiclass classifier, the RAM usage is significantly higher compared to all the other models trained with the MLP with one hidden layer.

In general, the results indicate that both the CPU and RAM usages are higher in the centralized framework, given that a larger MLP topology may be needed for improving the performance accuracy of the multiclass classifier, compared to the MLP applied for the distributed binary classifiers. It is important, however, to note that the distributed models have to be trained in parallel. Hence, the overall RAM usage of the distributed framework may become higher than the one required by the centralized framework. This higher RAM, however, is reserved for a shorter period of time (CPU usage), consequently mitigating the memory overhead caused by the parallel execution of the distributed models. These results are in general aligned with the fact that multiclass classification problems are harder to solve than their binary decompositions, and hence a larger MLP topology may be needed for the multiclass classifiers, at the expense of higher CPU and RAM requirements [33–35].

As the number of epochs required for model convergence is small for both the centralized and distributed models, the CPU usage does not significantly vary in these two frameworks, given that the MLP with one hidden layer is applied. For the larger MLP (applied to the centralized framework), the CPU usage significantly increases. However, as the number of epochs may be significantly larger in real datasets (e.g., the observed data may be noisy, and the nonlinear effects may render the problem harder), we also investigate the impact of both frameworks on the CPU metric by performing a number of simulations in which we assume that more epochs are required. Specifically, we gradually increase the number of required epochs and evaluate the CPU usage in the centralized and distributed frameworks. Figure 12 illustrates these results. Note that the RAM usage is not affected by the number of epochs. In fact, according to the Adam optimization algorithm applied for model training, the model parameters are updated after each epoch, and the same computations are performed in each epoch; hence, the memory requirements must be the same for each epoch, i.e., the previous model parameters are not saved in memory, and there is no additive memory overhead as the number of epochs increases.

Figure 12 illustrates the CPU usage (in seconds) versus the number of epochs for both centralized and distributed frameworks. For the distributed classifier, we have chosen to examine the model of the slice type $k=5$ for $K=5$ that is associated with the largest number of patterns and hence requires higher CPU usage. For the centralized case, the multiclass classifier with six classes is examined ($K=5$). For both the centralized and distributed frameworks, the MLP with one hidden layer is chosen in order to better illustrate that the CPU usage of the centralized framework tends to increase faster as the number

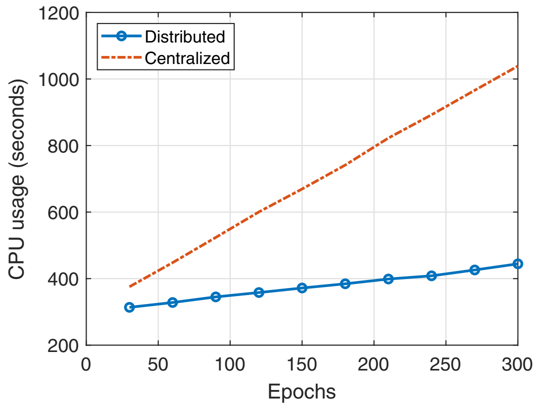

Fig. 12. CPU usage versus the number of epochs for the distributed and centralized frameworks for $K=5$.

of epochs increases (Fig. 12), even if the same MLP topology is applied. In general, a larger number of epochs means more iterations of the optimization algorithm (model updates), consequently increasing the CPU usage (training time). The CPU usage of the distributed framework greatly outperforms the centralized framework, due mainly to the smaller dataset it has to process at each epoch and the simpler arithmetic operations it has to perform. Note that the CPU usage refers to both the training and testing (inference) procedure. However, the inference time is negligible for both the distributed and centralized frameworks. Specifically, the inference time is on the order of milliseconds per test pattern (lightpath) for both frameworks, and thus the overhead in the CPU usage is caused mainly due to the training procedure.

Regarding the RAM usage, in general, memory is required to store input data, weight parameters, and activations. Furthermore, training an NN requires a large number of simple arithmetic operations, of which the intermediate calculations need to be stored in memory [50]. As previously mentioned, the number of epochs does not have an impact on the RAM usage, as model updates are performed per epoch, and the previous model parameters (or the gradients) do not need to be stored after each model update. However, as the MLP topology significantly affects the RAM usage, i.e., more hidden layers and units mean more weight parameters, activations, and arithmetic operations, we examined the RAM usage as the MLP topology increases in numbers of hidden layers and units. These results serve to show that in the centralized framework, which may require a larger MLP topology than its distributed decompositions, the RAM usage may significantly increase.

Figure 13 illustrates the RAM usage versus MLP topology in numbers of hidden layers and units. The results correspond to the multiclass classifier consisting of six classes. Note, however, that the RAM usage is negligibly affected by the size of the training dataset and the number of classes, at least according to the size of the datasets and the number of classes examined in our QoT estimation problem (this is clearly shown in the RAM usage results of Tables 2, 3, 5, and 7). According to Fig. 13, as the number of hidden layers and units increases, the RAM usage increases exponentially (note that the fluctuations in Fig. 13 are due mainly to the fact that, for simplicity in the

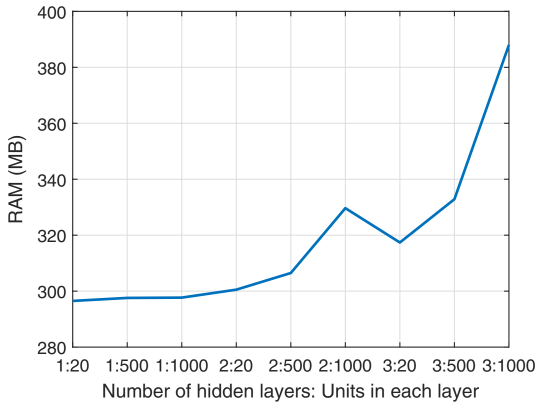

Fig. 13. RAM usage versus the number of hidden layers and units.

simulations, the number of units does not increase smoothly with the number of hidden layers). Hence, assuming that the centralized framework requires a larger MLP topology (in our results, the larger MLP multiclass classifier achieved better classification performance compared to the smaller MLP multiclass classifier, while the smaller MLP binary classifiers achieved an overall higher performance), the distributed framework has in general lower RAM requirements per slice type controller. Note, however, that in the distributed framework, several binary classifiers have to be trained in parallel, and hence the RAM requirement is additive to the number of diverse slice types, a trade-off between the higher accuracy achieved within considerably less CPU time and the higher RAM usage that may be required.

Overall, utilizing the distributed framework, less CPU and RAM are required for each slice type controller, essentially shifting functionalities from the central controller to the distributed controllers. One drawback of the distributed approach is the higher overall RAM usage that may be needed for training in parallel the various distributed models. This drawback, however, is mitigated by the considerably shorter time that this memory needs to be reserved for finding the distributed models. Investigating other metrics for the distributed versus the centralized case, such as energy consumption and communication overhead, is out of the scope of this work. Nevertheless, it is important to note that in the distributed approach, extra communication bandwidth will be needed to send the models' parameters to the central component to execute the QoT-aware multi-slice provisioning phase, as well as to send the patterns to their respective local controllers. The latter situation may occur if the patterns cannot be extracted locally (e.g., the BER ground truth of a pattern is not locally monitored, and this information must be communicated to the corresponding local controller).

Examining and comparing the performance of both the centralized and distributed approaches with the use of real data constitute an interesting future direction. In this work, in the absence of real data, we used synthetic data generated by a Q-tool [28], a common practice in the QoT estimation literature [10]. It is true, however, that real data may be noisier, thus affecting the performance of both frameworks. In such a case, and as shown in the ML literature, the MLP can be set up to successfully handle noise, i.e., by adding noise to the training input [51]. Also, it has been recently shown through field trials that NNs can achieve high QoT estimation accuracy [39]. Even though both the centralized and distributed approaches are expected to be negatively affected by the use of real data (and hence the MLPs will need to be adjusted accordingly), comparatively, it is expected that still the distributed approach will outperform the centralized approach in accuracy, accuracy per class, CPU usage, and RAM requirements per slice type, especially as the number of diverse slice types increases. Also, it is well accepted in the ML literature that multiclass classification problems are in general harder to solve (and more computationally expensive) than their alternative binary decompositions, especially as the number of classes increases. Hence, it is a common practice in the ML literature to reduce the multiclass problem by binary decompositions [33–35].

The reader should also note that in this work, for simplicity, and in order for the centralized and distributed frameworks to be more easily compared, we opted for the same ML method to be applied to every distributed and centralized controller. However, given the diversity of the datasets (i.e., number of patterns, imbalanced classes) or under real datasets that may be noisier, it is possible that the application of different ML methods may result in a higher classification performance, possibly at the expense of higher RAM and CPU usage. Indicatively, recurrent NNs (RNNs) and long short-term memory (LSTM) networks could achieve a better classification accuracy but have higher computational and memory requirements than the MLPs [52]. Since the simpler MLP approach applied in this work returned classifiers of an overall high accuracy, we opted not to examine other, more computationally and memory-intensive ML methods. The investigation of different ML methods according to their accuracy and RAM and CPU usage constitutes, however, an interesting future direction.

8. NETWORK PERFORMANCE EVALUATION

In this section, we examine the performance evaluation of the algorithms described in Section 6. The aim is to demonstrate the improvement in the network performance when the multi-slice QoT-aware RSA scheme is applied, instead of the conventional single-slice QoT-aware RSA approach. Note that both schemes assume the presence of pre-trained QoT model/s integrated into the RSA heuristics. As the distributed framework returned QoT models of higher accuracy compared to the centralized approach, in this section, we perform simulations only according to the distributed QoT models. Note, however, that simulations according to the centralized model are expected to yield similar results, as both the distributed and centralized models achieved sufficient RSA accuracy (Tables 8 and 9).

For the simulations, an EON is implemented using the Telefonica network topology (Fig. 7) with a spacing of 25 GHz and a 16 Gbaud rate per slot, which results in a total of 160 spectrum slots per link in the network. Each connection request, $C_n^k = \{s, d, B^k\}$, is again generated as described in Section 7.A.

For the set of BER requirements $\mathbf{B}$, five different cases were examined. For each case, we considered the presence of different slice types that vary in both their number (K) and their

Table 10. B Sets for Each Value of K

K	$\mathbf{B} = \{B^k\}_{k=1}^{K}$
1	$\{10^{-8}\}$
2	$\{10^{-8}, 10^{-7}\}$
3	$\{10^{-8}, 10^{-7}, 10^{-6}\}$
4	$\{10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}\}$
5	$\{10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}\}$

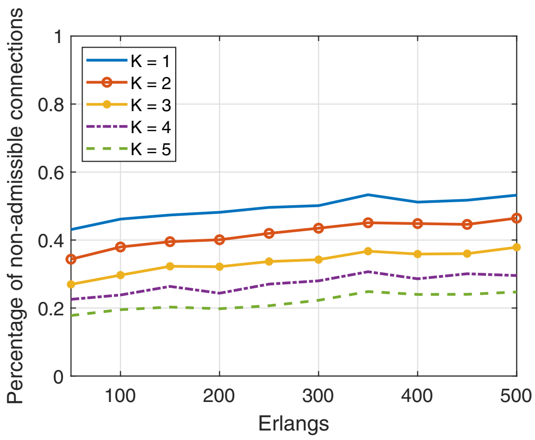

Fig. 14. Percentage of non-admissible connections versus different network loads for $K = 1$–5.

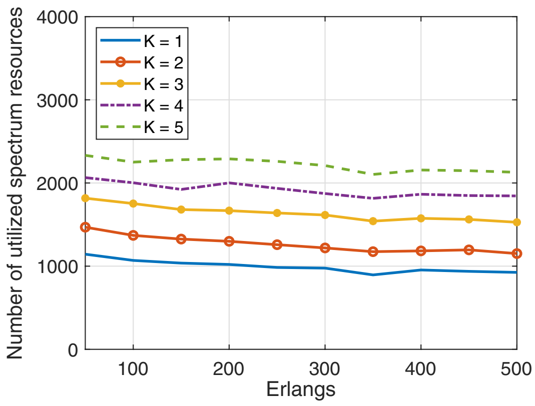

Fig. 15. Number of utilized spectrum resources versus different network loads for $K = 1$–5.

BER requirements assumed in each **B** set. The **B** sets considered for each different K value are given in Table 10. According to Table 10, when $K = 1$, a single slice is present with a single BER requirement. Hence, the $K = 1$ case corresponds to the benchmark approach, where a single QoT model of the lowest possible BER requirement is assumed during the connection provisioning phase. The rest of the K values in Table 10 correspond to the multi-slice QoT-aware RSA heuristic.

Five simulation runs were performed for each value of K. In each simulation run, 500 connection requests were generated according to the Poisson process with exponentially distributed holding times. The results were averaged over all the runs and are illustrated in Figs. 14 and 15. In particular, Fig. 14 illustrates the percentage of non-admissible connections for different network loads for all the different slice type cases, while Fig. 15 illustrates the spectrum utilization for different network loads, calculated over all established connections.

Results in Fig. 14 clearly show that the multi-slice QoT-aware RSA approach ($K = 2$ to 5) significantly outperforms the single-slice QoT-aware RSA approach ($K = 1$), especially as the number of diverse BER requirements considered increases. Specifically, as K increases, the percentage of non-admissible connections decreases considerably. This is due to the fact that in the multi-slice scenario, connections of higher BERs are admitted, provided that their BER requirement is met. On the contrary, in the single-slice approach, all connections will be rejected if their QoT does not meet the lowest BER requirement (even though for some of these connections, their actual BER requirement may be higher).

Consequently, and as shown in Fig. 15, the single-slice approach ($K = 1$) will lead to underutilization of the network resources. Specifically, Fig. 15 clearly shows that as the number of diverse slice types increases, the network resources are better utilized, as now more connections of diverse BER requirements can be admitted into the network. Note that in this work, without loss of generality, in order to better showcase the impact of the diverse BER requirements on network performance, the lowest BER requirement is set to 10^{-8} (i.e., a low threshold). Therefore, the number of non-admissible connections of the single-slice approach is impractically high. In particular, the choice of the lowest BER requirement was made according to the range of BERs generated by the Q-tool [28]. Variations to the QoT-aware RSA heuristic applied in this work could slightly improve the results in Figs. 14 and 15 (e.g., applying the κ-shortest paths algorithm [53] instead of Dijkstra's algorithm). However, the comparative results of all the K cases examined are expected to have the same tendency.

9. CONCLUSION

In this work, different QoT estimation frameworks based on centralized as well as distributed approaches are examined for sliceable optical networks, where connections with different BER requirements can be supported. The centralized QoT estimation framework was formulated as a multiclass classifier, whereas the distributed QoT estimation framework was formulated as a set of binary classifiers. An MLP was applied for training both the centralized and the distributed QoT models. Performance results obtained for a variety of different network parameters show that for the centralized QoT model, as the number of diverse BER requirements considered increases, the accuracy per class decreases (drops to 79%). On the other hand, for the distributed QoT models that are independent of the number of diverse BER requirements, high accuracy (above 91%) is achieved for both classes of interest. The RSA accuracy is sufficiently high for both frameworks, indicating that a low per class accuracy for the centralized framework may eventually not affect the QoT-aware RSA decisions. An extra step is, however, required for the centralized framework to validate the RSA accuracy, whereas for the distributed framework, these results are readily available after model training and testing. Furthermore, the training time required for each distributed QoT model is significantly lower compared to the centralized case, as for each distributed QoT model, only the lightpaths with the same BER requirement are taken into consideration during training. Regarding the RAM usage, it was shown that each distributed classifier requires less RAM than the centralized classifier. Importantly, it was also shown that the network performance is significantly improved when the BER requirements of each diverse slice are considered for the multi-slice QoT-aware RSA framework, an indicator that the multi-slice QoT-aware RSA framework avoids connection overprovisioning that occurs when the conventional single-slice QoT-aware RSA framework is utilized.

Future work includes the development of a framework that automatically provisions the required number and type of distributed controllers for optimizing network resources in environments that dynamically change over time. Comparing the developed frameworks according to their energy consumption, communication, and control overheads also constitute interesting future directions.

Funding. European Union's Horizon 2020 Framework Programme (739551–KIOS CoE); Republic of Cyprus Directorate General for European Programmes, Coordination and Development.

REFERENCES

1. 5G PPP Architecture Working Group, "View on 5G architecture" 2017.
2. S. Kavanagh, "What is network slicing?" 2018, https://5g.co.uk/guides/what-is-network-slicing/.
3. I. Scales, "Nokia claims network slicing for the fixed network," 2018, https://www.telecomtv.com/content/fixed-access/nokia-claims-network-slicing-for-the-fixed-network-32703/.
4. A. Farrel, "Service function chaining (SFC) and network slicing in backhaul and metro networks in support of 5G," in *20th International Conference on Transparent Optical Networks (ICTON)* (2018).
5. SNS Telecom and IT, "SON (self-organizing networks) in the 5G era: 2019–2030—Opportunities, Challenges, Strategies & Forecasts," 2018.
6. A. Mayoral, R. Vilalta, R. Casellas, R. Martinez, and R. Muñoz, "Multi-tenant 5G network slicing architecture with dynamic deployment of virtualized tenant management and orchestration (MANO) instances," in *42nd European Conference on Optical Communication* (2016).
7. M. R. Raza, C. Natalino, A. Vidal, M. A. S. Santos, P. Öhlen, L. Wosinska, and P. Monti, "Demonstration of resource orchestration using big data analytics for dynamic slicing in 5G networks," in *European Conference on Optical Communication (ECOC)* (2018).
8. R. Alvizu, S. Troia, V. M. Nguyen, G. Maier, and A. Pattavina, "Network orchestration for dynamic network slicing for fixed and mobile vertical services," in *Optical Fiber Communications Conference and Exposition (OFC)* (2018).
9. J. de la Mata, I. de Miguel, R. J. Duran, N. Merayo, S. K. Singh, A. Jukan, and M. Chamania, "Artificial intelligence (AI) methods in optical networks: a comprehensive survey," Opt. Switching Netw. **28**, 43–57 (2018).
10. F. Musumeci, C. Rottondi, A. Nag, I. Macaluso, D. Zibar, M. Ruffini, and M. Tornatore, "An overview on application of machine learning techniques in optical networks," IEEE Commun. Surv. Tutorials **21**, 1383–1408 (2019).
11. X. Chen, R. Proietti, H. Lu, A. Castro, and S. J. B. Yoo, "Knowledge-based autonomous service provisioning in multi-domain elastic optical networks," IEEE Commun. Mag. **56**(8), 152–158 (2018).
12. R. Alvizu, S. Troia, G. Maier, and A. Pattavina, "Matheuristic with machine-learning-based prediction for software-defined mobile metro-core networks," J. Opt. Commun. Netw. **9**, D19–D30 (2017).
13. F. Morales, M. Ruiz, L. Gifre, L. M. Contreras, V. López, and L. V. Esteban, "Virtual network topology adaptability based on data analytics for traffic prediction," J. Opt. Commun. Netw. **9**, A35–A45 (2017).
14. T. Panayiotou, K. Manousakis, S. P. Chatzis, and G. Ellinas, "On learning bandwidth allocation models for time-varying traffic in flexible optical networks," in *International Conference on Optical Network Design and Modeling (ONDM)* (2018).
15. T. Panayiotou, K. Manousakis, S. P. Chatzis, and G. Ellinas, "A allocation framework with QoS considerations for EONs," J. Lightwave Technol. **37**, 1853–1864 (2019).
16. T. Panayiotou, S. P. Chatzis, and G. Ellinas, "Leveraging statistical machine learning to address failure localization in optical networks," J. Opt. Commun. Netw. **10**, 162–173 (2018).
17. B. Shariati, M. Ruiz, J. Comellas, and L. V. Esteban, "Learning from the optical spectrum: failure detection and identification," J. Lightwave Technol. **37**, 433–440 (2019).
18. S. Shahkarami, F. Musumeci, F. Cugini, and M. Tornatore, "Machine-learning-based soft-failure detection and identification in optical networks," in *Optical Fiber Communication Conference (OFC)* (2018).
19. B. Yan, Y. Zhao, Y. Li, X. Yu, J. Zhang, and H. Z. Yilin, "First demonstration of imbalanced data learning-based failure prediction in self-optimizing optical networks with large scale field topology," in *Asia Communications and Photonics Conference (ACP)* (2018).
20. C. Natalino, M. Schiano, A. Di Giglio, L. Wosinska, and M. Furdek, "Field demonstration of machine-learning-aided detection and identification of jamming attacks in optical networks," in *European Conference on Optical Communication (ECOC)* (2018).
21. M. Bensalem, S. K. Singh, and A. Jukan, "On detecting and preventing jamming attacks with machine learning in optical networks," arXiv:1902.07537v2 [cs.NI] (2019).
22. T. Panayiotou, S. P. Chatzis, and G. Ellinas, "Performance analysis of a data-driven quality-of-transmission decision approach on a dynamic multicast-capable metro optical network," J. Opt. Commun. Netw. **9**, 98–108 (2017).
23. T. Panayiotou, G. Savva, B. Shariati, I. Tomkos, and G. Ellinas, "Machine learning for QoT estimation of unseen optical network states," in *Optical Fiber Communications Conference and Exhibition (OFC)* (2019).
24. R. Morais and J. Pedro, "Machine learning models for estimating quality of transmission in DWDM networks," J. Opt. Commun. Netw. **10**, D84–D99 (2018).
25. J. Mata, I. de Miguel, R. J. Durán, J. C. Aguado, N. Merayo, L. Ruiz, P. Fernández, R. M. Lorenzo, E. J. Abril, and I. Tomkos, "Supervised machine learning techniques for quality of transmission assessment in optical networks," in *20th International Conference on Transparent Optical Networks (ICTON)* (2018).
26. C. Rottondi, L. Barletta, A. Giusti, and M. Tornatore, "Machine-learning method for quality of transmission prediction of unestablished lightpaths," J. Opt. Commun. Netw. **10**, A286–A297 (2018).
27. W. Fawaz, B. Daheb, O. Audouin, M. Du-Pond, and G. Pujolle, "Service level agreement and provisioning in optical networks," IEEE Commun. Mag. **42**(1), 36–43 (2004).
28. B. Shariati, A. Mastropaolo, P. Diamantopoulos, J. M. Rivas-Moscoso, D. Klonidis, and I. Tomkos, "Physical-layer-aware performance evaluation of SDM networks based on SMF bundles, MCFs, and FMFs," J. Opt. Commun. Netw. **10**, 712–722 (2018).
29. T. Panayiotou, G. Ellinas, N. Antoniades, and A. Hadjiantonis, "Impairment-aware multicast session provisioning in metro optical networks," Comput. Netw. **91**, 675–688 (2015).
30. G. Savva, G. Ellinas, B. Shariati, and I. Tomkos, "Physical layer-aware routing, spectrum, and core allocation in spectrally-spatially flexible optical networks with multicore fibers," in *IEEE International Conference on Communications (ICC)* (2018).
31. M. Klinkowski and K. Walkowiak, "Impact of crosstalk estimation methods on the performance of spectrally and spatially flexible

optical networks," in *20th International Conference on Transparent Optical Networks (ICTON)* (2018).
32. T. Panayiotou, G. Savva, I. Tomkos, and G. Ellinas, "Centralized and distributed machine learning-based QoT estimation for sliceable optical networks," in *IEEE Global Communications Conference* (2019).
33. D. Silva-Palacios, C. Ferri, and M. J. Ramírez-Quintana, "Improving performance of multiclass classification by inducing class hierarchies," Procedia Comput. Sci. **108**, 1692–1701 (2017).
34. A. C. Lorena, A. C. P. L. F. de Carvalho, and J. M. P. Gama, "A review on the combination of binary classifiers in multiclass problems," Artif. Intell. Rev. **30**, 19–37 (2008).
35. M. Galar, A. Fernández, E. Barrenechea, H. B. Sola, and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: experimental study on one-vs-one and one-vs-all schemes," Pattern Recognit. **44**, 1761–1776 (2011).
36. S. Yan, R. Nejabati, and D. Simeonidou, "Data-driven network analytics and network optimisation in SDN-based programmable optical networks," in *International Conference on Optical Network Design and Modeling (ONDM)* (2018).
37. C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer-Verlag, 2006), chap. 5.
38. R. Proietti, X. Chen, A. Castro, G. Liu, H. Lu, K. Zhang, J. Guo, Z. Zhu, L. Velasco, and S. J. B. Yoo, "Experimental demonstration of cognitive provisioning and alien wavelength monitoring in multi-domain EON," in *Optical Fiber Communication Conference (OFC)* (2018).
39. Z. Gao, S. Yan, J. Zhang, M. Mascarenhas, R. Nejabati, Y. Ji, and D. Simeonidou, "ANN-based multi-channel QoT-prediction over a 563.4-km field-trial testbed," J. Lightwave Technol. (to be published).
40. P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," in *Proc. ICLR* (2018).
41. D. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proc. ICLR* (2015).
42. C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer-Verlag, 2006), chap. 4.
43. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Dijkstra's algorithm," in *Introduction to Algorithms*, 3rd ed. (MIT, 2009) , section 24.3.
44. W. Mo, Y.-K. Huang, S. Zhang, E. Ip, D. C. Kilper, Y. Aono, and T. Tajima, "ANN-based transfer learning for QoT prediction in real-time mixed line-rate systems," in *Optical Fiber Communication Conference (OFC)* (2018).
45. D. Azzimonti, C. Rottondi, and M. Tornatore, "Using active learning to decrease probes for QoT estimation in optical networks," in *Optical Fiber Communication Conference (OFC)* (2019).
46. S. Karsoliya, "Approximating number of hidden layer neurons in multiple hidden layer BPNN architecture," Int. J. Eng. Trends Technol. **3**, 714–717 (2012).
47. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," J. Mach. Learn. Res. **15**, 1929–1958 (2014).
48. A. Tharwat, "Classification assessment methods," in *Applied Computing and Informatics* (2018).
49. S. Becker, Y. Zhang, and A. A. Lee, "Geometry of energy landscapes and the optimizability of deep neural networks," arXiv:1808.00408v1 [cond-mat.dis-nn] (2018).
50. J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," arXiv:1812.02858v2 [cs.IT] (2019).
51. H. Noh, T. You, J. Mun, and B. Han, "Regularizing deep neural networks by noise: its interpretation and optimization," in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)* (2017).
52. N. M. Rezk, M. Purnaprajna, T. Nordström, and Z. Ul-Abdin, "Recurrent neural networks: an embedded computing perspective," arXiv:1908.07062v2 [cs.NE] (2019).
53. J. Y. Yen, "Finding the *K* shortest loopless paths in a network," Manage. Sci. **17**, 712–716 (1971).

Tania Panayiotou (M'09) received her degree in computer engineering and informatics from the University of Patras, Greece, in 2005 and her Ph.D. in computer engineering from the University of Cyprus, Cyprus, in 2013. She is currently a Research Fellow at the KIOS Research and Innovation Center of Excellence at the University of Cyprus. She has previously worked as an Associate Researcher in the Department of Electrical Engineering, Computer Engineering and Informatics of the Cyprus University of Technology, and as an Adjunct Lecturer in the Department of Electrical and Computer Engineering at the University of Cyprus and in the Department of Information and Communication Systems at the Open University of Cyprus. Dr. Panayiotou is a Member of IEEE. She has co-authored more than 28 articles, conference papers, and book chapters. She has received the best paper award in the ONDM'17 conference. Her research interests focus on optical networks as well as transportation networks.

Giannis Savva (GSM'17) received his B.Sc. degree in electrical engineering from the Department of Electrical and Computer Engineering at the University of Cyprus in 2017. He is currently a Ph.D. Candidate in the Department of Electrical and Computer Engineering and a Graduate Research Assistant at the KIOS Research and Innovation Center of Excellence, University of Cyprus, Nicosia, Cyprus. Mr. Savva is a Graduate Student Member of IEEE. His research interests are in the areas of telecommunications, resource allocation algorithms in spectrally-spatially flexible optical networks (SS-FONs), network planning, network coding, and physical layer security in optical networks.

Ioannis Tomkos (M'96—SM'04—F'18) is a Professor of Optical Communications in the Department of Electrical and Computer Engineering (ECE) of the University of Patras. In the past he was a Full Professor and Research Director at the Athens Information Technology Center-AIT, Greece; a Chair of Excellence Professor (Cátedras de Excelencia) at the University Carlos III, Spain; an Adjunct Professor in the College of Optical Sciences of the University of Arizona, USA; an Adjunct Research Fellow in the ECE Department of the University of Cyprus; an Adjunct Faculty Member in the Information Networking Institute of Carnegie-Mellon University, USA; a Senior Scientist at Corning Inc, USA; and a Research Assistant at the University of Athens, Greece. At AIT he founded and served as the Head of the "High Speed Networks and Optical Communication (NOC)" group that was involved in over 25 EU-funded research projects within which Prof. Tomkos served as Principal Investigator having a consortium-wide leading role. Prof. Tomkos, together with his colleagues and students, has authored over 650 peer-reviewed archival articles, including over 150 journal/magazine/book publications. His published work has received over 9600 citations and his h-factor is 47 (as of Feb. 2020 based on data from Google Scholar). For his academic achievements, he was elected as a Fellow of IEEE "for contributions in Dynamic Optical Networks" (2018), Fellow of the IET (2010), and a Fellow of OSA (2012).

Georgios Ellinas (M'98—SM'07) holds a B.S., M.Sc., M.Phil., and a Ph.D. in electrical engineering from Columbia University. He is a Professor and the Chair of the Department of Electrical and Computer Engineering and a founding member of the KIOS Research and Innovation Center of Excellence at the University of Cyprus. Previously, he served as an Associate Professor of Electrical Engineering at the City College of New York (2002–2005), a Senior Network Architect at Tellium Inc. (2000–2002), and a Research Scientist/Senior Research Scientist at Bell Communications Research (Bellcore) (1993–2000). Prof. Ellinas is a Fellow of the IET (2019), a Senior Member of IEEE and OSA, and a Member of ACM and the Marie Curie Fellows Association (MCFA). He has co-authored/co-edited 4 books on optical networks as well as more than 230 archived articles, conference papers, and book chapters, and he is the holder of 30 patents on optical networking. His research interests are in optical/telecommunication networks, transportation networks, IoT, and critical infrastructure systems.