A Data-Driven QoT Decision Approach for Multicast Connections in Metro Optical Networks

Conference Paper · May 2016

DOI: 10.1109/ONDM.2016.7494074

3 authors:

Tania Panayiotou

18 PUBLICATIONS **52** CITATIONS

SEE PROFILE

Georgios Ellinas

University of Cyprus

191 PUBLICATIONS **2,049** CITATIONS

SEE PROFILE

Sotirios P Chatzis

Cyprus University of Technology

65 PUBLICATIONS **533** CITATIONS

SEE PROFILE

A Data-Driven QoT Decision Approach for Multicast Connections in Metro Optical Networks

Tania Panayiotou, Georgios Ellinas
KIOS Research Center for Intelligent Systems and Networks,
Department of Electrical and Computer Engineering
University of Cyprus, Cyprus
Email:{panayiotou.tania, gellinas}@ucy.ac.cy

Sotirios P. Chatzis
Department of Electrical Engineering,
Computer Engineering, and Informatics,
Cyprus University of Technology, Cyprus
Email: sotirios.chatzis@cut.ac.cy

***Abstract*—A data-driven technique for analyzing Quality-of-Transmission (QoT) data of previously established connections is proposed for accurately deciding the QoT of the newly arriving multicast requests in metro optical networks. The proposed approach is self-adaptive, it is a function of data that are independent from the physical layer impairment (PLIs) and thus does not require specific measurement equipment, and it does not assume the existence of a system with extensive processing and storage capabilities. It is also fast in processing new data, and fast in finding a near-accurate QoT model provided that such a model exists. The proposed technique can replace the existing Q-factor models that are not self-adaptive, they are a function of the PLIs, and their evaluation requires time-consuming simulations, lab experiments, specific measurement equipment, and considerable human effort. The proposed data-driven QoT approach is based on the utilization of a feed-forward neural network that is trained on a dataset previously generated from a known Q-factor model. The dataset fed to the neural network is represented in a way that specifically describes the QoT of the multicast connections requesting to be established in the network but it is independent from the PLIs. The validity of the proposed approach is examined for two distinct networks, exhibiting a high accuracy when compared to the results of the Q-factor model utilized for generating the QoT data.**

I. Introduction

Recent advances in optical networks, that are expected to support traffic that will be heterogeneous in nature (i.e., capable of supporting unicast and multicast traffic), have made bandwidth-intensive point-to-multipoint (P2MP) applications widely popular. However, if such applications are offered in transparent optical networks, the physical layer impairments must be taken into consideration to ensure that the signal can be correctly detected at the receiver [1]. Several works exist in the literature that use different representations for modeling the most important physical layer effects that can accumulate in a transparent optical network, such as ASE noise, crosstalk, optical filter concatenation, and polarization mode dispersion (PMD) amongst others [1], [2]. One approach for the modeling of the physical layer is based on the physical path Q-factor that is subsequently used to calculate the Bit Error Rate (BER) of the system, a parameter that is difficult to evaluate upfront. In these representations worst-case budget values are usually included for accounting for the physical layer effects that are difficult to be accurately evaluated (e.g., crosstalk, PMD, polarization depended gain/loss, etc.) and have been previously evaluated statistically [3]. The latter approach involves time-/frequency-domain Monte Carlo simulations on true measurements for each one of the PLIs that are present in a transparent optical network [3]. Although these representations are valuable at the engineering and performance evaluation stages of a network, upon network changes the aforementioned time-consuming procedure needs to be repeated for updating the Q-factor model.

This work proposes the utilization of a time-efficient state-of-the-art machine learning technique for analyzing QoT data of previously established multicast connections aiming at accurately deciding the QoT of newly arriving requests. The advantages of the proposed approach over the existing Q-factor models are several; the proposed approach is self-adaptive, it is independent from the PLIs, it is fast in finding an accurate model (if such a model exists), it is fast in processing new data, and it does not assumes the existence of a system with extensive processing and storage capabilities. In particular, a feed-forward neural network from the context of pattern recognition is used that is reported to attain the above objectives [4]. The neural network is trained on a QoT dataset generated from the Q-factor formula described in detail in [2] and based on the analytical Q-budgeting approach proposed in [3]. The Q-factor model utilized for generating the data was developed for a multicast-capable metro network that considers multicast-capable node architecture/engineering for a network that spans a metropolitan area (i.e., utilizing an average distance of 80km between network nodes). The data generated from the Q-factor model were represented in a way that is independent from the PLIs but yet describes the QoT of each connection requesting to be established in the multicast-capable network. The accuracy of the proposed approach is evaluated by comparing the QoT decisions of the proposed approach to the QoT decisions of the Q-factor approach (the reader should note that no other Q-factor model, other than the one utilized in this work, exists in the literature regarding multicast connections).

The rest of the paper is outlined as follows. Section II describes the related existing work in terms of data-driven QoT decisions, while Section III describes the problem statement and provides an overview of the proposed approach. Subsequently, Section IV provides the problem formulation, the neural network training, as well as the QoT decision algorithm and Section V describes the training data and the selection of the features utilized to reach the QoT decision. Performance results are presented in Section VI, while Section VII provides some concluding remarks.

II. Related Work

Related works exist in the literature that examine the inferential QoT framework by either focusing on designing a Software Defined Network (SDN) platform capable of supporting data-driven QoT decisions or by proposing data-driven approaches for accurate QoT decisions [5], [6], [7], [8]. Specifically, authors in [5] proposed a transport SDN architecture, and presented new data for devices, network elements, and SDN applications, in order to enable optical networks to support new services and virtualization with flexibility and scalability. Experimental results were shown, demonstrating optical network self-adaptation for sustaining QoT in the advent of optical impairments. In [5], only point-to-point connections were considered and an Optical Signal to Noise Ratio (OSNR) monitoring scheme was utilized for keeping track of the physical impairments. However, no specific machine learning techniques were proposed.

Addressing the problem from a different point of view, authors in [6] explored the benefits of utilizing data-driven models for QoT decisions. However, in [6], only point-to-point connections were assumed and also no specific machine learning approaches were proposed. This work was extended in [7], where a data-driven QoT estimator was proposed for classifying lightpaths into high or low quality categories in impairment-aware wavelength-routed optical networks. In particular, the technique presented was based on Case-Based Reasoning (CBR), an artificial intelligence technique which solves new problems by exploiting previous experiences, which are stored in a knowledge database. Finally, in [8] a Gaussian Noise (GN) model was proposed which is able to estimate, quickly and accurately, the OSNR of the optical channels in uncompensated coherent transmission systems. However, in this work, no specific network scenarios were addressed. In summary, for all aforementioned works, only point-to-point connections were considered; to the best of our knowledge, this is the first time that a self-adaptive approach for QoT decisions regarding multicast connections is examined.

III. Problem Statement and Approach Overview

The existing QoT model for multicast connections [2], [9], is merely a function of the physical layer impairments and is based on the Q-factor modeling approach presented in [3]. The Q-factor model was evaluated with time-consuming Monte Carlo simulations that are based on stochastic numerical sampling from distributions, and each PLI (e.g., distortion-induced penalty due to filter concatenation, crosstalk, ASE noise, polarization mode dispersion, etc.) was evaluated by conducting experiments in the lab with the appropriate measurement equipment. A detailed analysis for the evaluation of each impairment and the procedure followed can be found in [3] and references therein. Although the existing analytical Q-factor model is valuable for evaluating the QoT of a connection prior to its establishment, upon network changes (i.e., network upgrades, equipment repairs, equipment replacement, ageing, etc.) the time-consuming procedure described in [3] must be conducted all over again. This, however, entails the usage of specific equipment, lab measurements, considerable human effort, and a process for deciding whether model re-evaluation is really necessary or not. Thus, in this work, the focus is on finding a model that is self-adaptive, it does not require lab measurements or specific equipment, it is time efficient, and it does not require the existence of a system with extensive processing and storage capabilities.

For finding such a model, a state-of-the-art feed-forward neural network is utilized, as feed-forward neural networks have been reported to be fast in model evaluation, fast in processing new data, result in compact models (as they require only a few training parameters), are adaptive during training, and also exhibit a high generalization performance. A detailed discussion on feed-forward neural networks and their advantages over other optimization methods in the context of pattern recognition can be found in Chapter 3 of [4].

The general framework of the proposed approach, that entails the utilization of the neural network can be briefly described as follows:

1) Data from the analytical Q-factor model are generated (or real QoT data are utilized if these are available).
2) These data are represented in a vector form that is independent from the PLIs but capable of describing the QoT of the connections requesting to be established in the network of interest.
3) The feed-forward neural network is trained on a training dataset
4) The accuracy of the neural network model is validated on a dataset other than the training dataset.

For training purposes an on-line procedure [10] can be used, instead of a batched procedure, in which patterns are sequentially fed into the neural network allowing each time for model updates. According to the on-line training procedure, the model can be updated sequentially in the evolving network with insignificant processing overhead. Thus, the technique is self-adaptive, while re-evaluation decisions are not really necessary. It is interesting to note that the above procedure can be used for any network scenario, provided that the QoT data are represented in a way that is independent from the PLIs but reflects the signal quality of the connections requesting to be established in the network of interest.

IV. Problem Formulation, Neural Network Training, and QoT Decision Algorithm

A. Problem Formulation

The data-driven QoT problem is treated as a binary classification problem in which the goal is to take an input vector $\boldsymbol{x}$ and to assign it to a discrete value y, where $y \in \{0, 1\}$ [4]. In this work the set $\mathcal{D} = \{(\boldsymbol{x}(\boldsymbol{j}), y(j))|j = 1, ..., n\}$ is defined to be the training dataset in which pattern j represents lightpath j, $y(j) \in 0, 1$ is the target value, and $\boldsymbol{x(j)^T} = [x_1(j), x_2(j), x_3(j), x_4(j), x_5(j)]$. Specifically, $x_1(j)$ corresponds to the nominal path length of j, $x_2(j)$ to the number of erbium doped fiber amplifiers (EDFAs) in j, $x_3(j)$ denotes the nominal maximum link length of j, and $x_4(j)$ denotes the degree of the destination node in j. Note that $x_5(j)$ is the bias of the first layer of the neural networks that is set at -1.

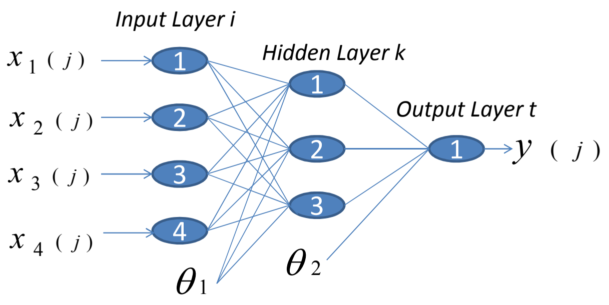

Fig. 1. Neural network with 4 input units, hidden layer k with 3 units, and one output.

Figure 1 illustrates the specific feed-forward neural network assumed in this work that consists of 4 input units in layer i, one hidden layer k with 3 units, and one output layer t with a single output unit. Note that the specific neural network was chosen after a few trials that examined the utilization of more hidden layers. The neural network that exhibited the best performance with respect to the QoT dataset of interest was chosen. The unknown parameters that are determined through the training procedure are denoted with w_{ik} and w_{kt}, where w_{ik} is the network weight parameter from unit i to unit k, and w_{kt} is the network weight parameter from unit k to unit t (not shown in Fig. 1). For training the neural network, the backpropagation BP algorithm [11] is utilized as described next.

B. Backpropagation Algorithm for Network Training

The BP algorithm is a gradient descent method that optimally learns the weights in a multi-layer neural network. For system initialization, small arbitrary weights are chosen. To accomplish learning, BP is successively adjusting the weights based on a set of input patterns and the corresponding set of desired output patterns. During this process, an input pattern is presented to the network and propagated forward through the network to determine the resulting signal at the output units. The error is evaluated by the difference between the actually resulting output signal and the predetermined desired output signal in each output unit. The error is then backpropagated through the network in order to adjust the weights. The learning process continues for a maximum number of iterations that is set a priori, or until the convergence of the squared error function. The error E function is defined as

$$E = \Sigma_{j=1}^{n} (y(j) - o(j))^2 \tag{1}$$

where n is the number of the input patterns, $y(j)$ is the expected output value for pattern j, and $o(j)$ is the output evaluated by the BP algorithm after the presentation of pattern j to the network.

When input pattern j is applied to the network, the activation of each unit is dynamically determined using the sigmoid function (2) as in our neural net we opt for a sigmoid nonlinearity.

$$s(z) = \frac{1}{1 + exp(-z)} \tag{2}$$

Thus, according to Fig. 1,

$$o_k(j) = s(\Sigma_{i=1}^{5} w_{ik} x_i(j)) \tag{3}$$

and

$$o(j) = s(\Sigma_{k=1}^{4} w_{kt} o_k(j)) \tag{4}$$

where $o_k(j)$ is the activation of unit k as a result of the application of pattern j, w_{ik} is the weight from unit i to unit k, and the bias for layer i is set to $\theta_1 = x_5(j) = -1$. Similarly, $o(j)$ is the activation of unit t as a result of the application of pattern j, w_{kt} is the weight from unit k to unit t, and the bias for layer k is set to $\theta_2 = o_4(j) = -1$.

Backpropagation then takes place to update all of the weights in the network according to

$$\Delta w_{kt} = \eta \delta_t(j) o(j), \quad \Delta w_{ik} = \eta \delta_k(j) o_k(j) \tag{5}$$

where η is the learning rate, $\delta_t(j)$ is the error signal for unit t, and $\delta_k(j)$ is the error signal for unit k after the presentation of pattern j to the network. The error signal $\delta_t(j)$ for output unit t is calculated from the difference between the target and actual value for that unit as

$$\delta_t(j) = (y(j) - o(j)) o(j) (1 - o(j)) \tag{6}$$

The error signal $\delta_k(j)$ for a hidden unit k is a function of the error signals of those units in layer t connected to unit k and the weights of those connections. Specifically,

$$\delta_k(j) = o_k(j)(1 - o_k(j)) d_t(j) w_{kt} \tag{7}$$

C. Training and QoT Decision Algorithm

For training the neural network, the aforementioned procedure is employed sequentially for each pattern. Doing so, the neural network is trained in an *on-line* fashion [10], instead of using the data set as a single batch, in order to avoid any scalability issues that batch training suffers from, especially when dealing with large training datasets, and to attain self-adaptation in the evolving network. In particular, after each input pattern is presented to the network, the error across the output unit is determined and backpropagated to update the weights. The next pattern is then presented and the process is repeated. The learning rate η is varied according to whether or not an iteration decreases the error for all patterns. If an update results in reduced total error or if the total error is increased only by 0.5%, η is multiplied by a factor $\alpha > 1$ for the next iteration. Otherwise, η is multiplied by a factor $\beta < 1$. In order to escape from local minima, if two sequential iterations fail to update the system and the total error is not zero, then η is randomly generated with $0 < \eta < 1$ while the small arbitrary weights are randomly reinitialized [12].

The learning process continues for a maximum number of iterations that is set a priori, or until the convergence of the squared error function. If the solution for the specific neural network of Fig. 1 is given by the two weight matrices W_{ik} and W_{kt} (W_{ik} is shown in Eq. 8 and $W_{kt}^T = [w_{11}, w_{21}, w_{31}, w_{41}]$), then the value $y(*)$ for a lightpath with data vector $\boldsymbol{x(*)^T} = [x_1(*), x_2(*), x_3(*), x_4(*), -1]$ is given by Algorithm 1.

$$W_{ik} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \\ w_{51} & w_{52} & w_{53} \end{bmatrix} \quad (8)$$

Algorithm 1 Data-Driven QoT Model

Input: W_{ik}, W_{kt}, and $\boldsymbol{x(*)}$
Output: $y(*)$

```
1:  $T_{thresh} \leftarrow s(-w_{41})$ where $w_{41} \in W_{kt}$
2:  $W_{kt}^{'T} = [w_{11}, w_{21}, w_{31}]$
3:  $o_k(*) = s(W_{ik}^T \boldsymbol{x(*)})$
4:  $o(*) = s(W_{kt}^{'T} o_k(*))$
5:  if $o(*) < T_{thresh}$ then
6:      $y(*) = 0$
7:  else
8:      $y(*) = 1$
9:  end if
10: return $y(*)$
```

V. Training Data and Feature Selection

For the generation of the training dataset $\mathcal{D} = \{(\boldsymbol{x(j)}, y(j)) | j = 1, ..., n\}$, a metropolitan area optical network was utilized and its characteristics correspond to Network A of Table I. Further, the Q-factor formulation [2] in combination with the network architecture/engineering assuming multicast-capable nodes with fixed TXs/RXs were assumed. Note that for a detailed analysis of the Q-modeling, node architecture, and node engineering considered, the reader is referred to previous work in [2], [9].

In order to generate the dataset $\mathcal{D}$, the k-Steiner Tree (k-ST) heuristic was developed that is an extension of the conventional Steiner Tree (ST) heuristic [13], calculating k multicast trees for each call. In particular, the k-ST heuristic in the m^{th} iteration calculates the m^{th} multicast tree by excluding from the available resources a link that is randomly chosen amongst the links utilized by the $(m-1)^{th}$ multicast tree. Excluded links are assigned weights that are several orders of magnitude greater than their original weights in order to allow the k-ST algorithm to create k different light-trees.

One thousand (1000) multicast calls were generated of random multicast group sizes, varying from 2 to 23 destinations. For each call, 10 STs were calculated and the Q-factor for every destination node was evaluated. Each light-tree was then decomposed to its constituent lightpaths for a total number of n lightpaths. Thus, n training patterns were generated. The Q-factor of each destination was compared to a Q-threshold and converted to a binary value $y(j)$, where j refers to pattern j; $y(j) = 1$ for Q-values above the Q-threshold, $y(j) = 0$ otherwise. Note that in this work 97% of the n patterns yield $y(j) = 1$ for a Q-threshold set to 9dB (corresponding to a BER of 10^{-15}).

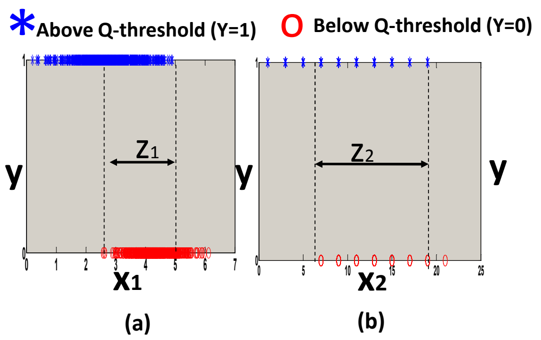

Fig. 2. Set of outputs $\boldsymbol{y}$ vs set of input patterns (a) $\boldsymbol{x_1}$, (b) $\boldsymbol{x_2}$. Respective uncertainty regions Z_1 and Z_2 are also shown.

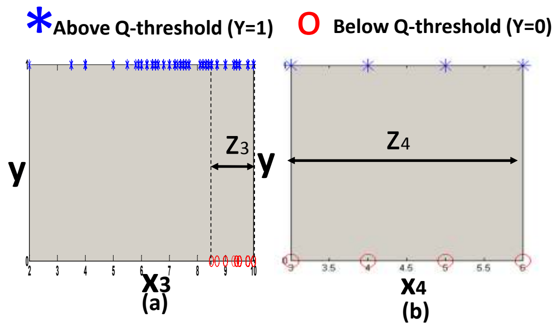

Fig. 3. Set of outputs $\boldsymbol{y}$ vs set of input patterns (a) $\boldsymbol{x_3}$, (b) $\boldsymbol{x_4}$. Respective uncertainty regions Z_3 and Z_4 are also shown.

The rest of this section describes how the data that effectively describe the established connections, and are independent from the PLIs, were selected. Figures 2 and 3 show how some of the data collected from each lightpath j were correlated to the signal quality of each lightpath. For brevity, let $\boldsymbol{y} = \{y(j)\}_{j=1}^n$ and $\boldsymbol{x_k} = \{x_k(j)\}_{j=1}^n$ $\forall k$, where $k = 1, 2, 3, 4$. Specifically, Fig. 2(a) illustrates $\boldsymbol{y}$ versus $\boldsymbol{x_1}$ and clearly shows that the lightpath length feature if utilized alone for QoT decisions, cannot lead to accurate QoT decisions due to the uncertainty region Z_1 denoted in Fig. 2(a) that includes mutual information for both levels in

$\boldsymbol{y}$. Specifically, according to the results found, $P(x_1(j) \in Z_1|y(j) = 0) = 0.9$, $P(x_1(j) \in Z_1|y(j) = 1) = 0.02$, and $P(x_1(j) \in Z_1) = 0.045$, which means that only 4.5% of the total generated lightpaths lie in region Z_1 but amongst them lie 90% of the lightpaths with a Q-value below the Q-threshold. Therefore, the path length is not a sufficient decision feature, as a large number of connections with unacceptable QoT are included in the uncertainty region Z_1. Similarly, Fig. 2(b) illustrates $\boldsymbol{y}$ versus $\boldsymbol{x_2}$ and according to the results obtained, $P(x_2(j) \in Z_2|y(j) = 0) = 0.99$, $P(x_2(j) \in Z_2|y(j) = 1) = 0.02$, and $P(x_2(j) \in Z_2) = 0.05$, while for Fig. 3(a), $P(x_3(j) \in Z_3|y(j) = 0) = 1$, $P(X_3(j) \in Z_3|y(j) = 1) = 0.03$, and $P(X_3(j) \in Z_3) = 0.06$. Finally for Fig. 3(b), $P(x_4(j) \in Z_4|y(j) = 0) = 1$, $P(X_4(j) \in Z_4|y(j) = 1) = 1$, and $P(X_4(j) \in Z_4) = 1$. According to the above, none of the $\boldsymbol{x_1}, \boldsymbol{x_2}$, $\boldsymbol{x_3}$, and $\boldsymbol{x_4}$ set of data can be considered alone for accurate QoT decisions. However, if a number of data are considered jointly, then two linearly separable sets may be possible. Fig. 4 clearly shows that the combined information from several data ($\boldsymbol{x_1}$, $\boldsymbol{x_2}$, $\boldsymbol{x_4}$ in this case), creates two distinct sets of lightpaths that seem to be linearly separable. However, as it is not possible to know whether the set of patterns is linearly separable, the BP method previously described is used that is capable of finding a separation that minimizes the number of incorrectly classified points, even if the set of patterns is not linearly separable.

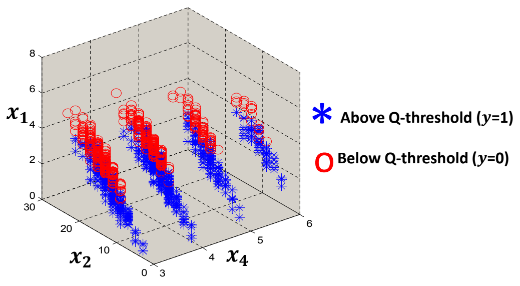

Fig. 4. Set of outputs $\boldsymbol{y}$ vs $\boldsymbol{x_1}$, $\boldsymbol{x_2}$, and $\boldsymbol{x_4}$ in three dimensions.

Apart from data features $\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}$, and $\boldsymbol{x_4}$, other data were also extracted as candidate input data features to the learning algorithm, such as the degree of the source node and the minimum/average link lengths of each lightpath. However, they were not considered in the BP algorithm, as they were either not found to contribute additional information or were found to be uncorrelated to the signal quality, at least for the network architecture/engineering under consideration. As an example, the splitting losses of source nodes are uncorrelated to the signal quality since in the engineering case considered variable optical attenuators are used to equalize the signals at the EDFAs just after the optical splitters.

VI. Performance Results

In order to examine the accuracy of the aforementioned method, the neural network of Fig. 1 was trained for two distinct networks, namely Network A and Network B, described in Table I. Note that Network B is given in detail in Table II while both networks were generated in such a way in order to meet metro network specifications (average distance between the nodes, network diameter, number of nodes, number of links) for which the Q-factor model is valid. The results for both networks are discussed next.

TABLE I
Network Statistics

Network	A	B
Number of Nodes	50	14
Number of Bidirectional Links	96	50
Average Distance (km)	60	67
Maximum Distance (km)	100	100
Minimum Distance (km)	29	20
Average Node Degree	3.92	7.15
Minimum Node Degree	3	4
Maximum Node Degree	6	10
Diameter (km)	305	160
Diameter (hops)	6	3

A. Results for Network A

For Network A, the neural network of Fig. 1 was trained according to the BP method considering the data set $\mathcal{D} = \{(\boldsymbol{x(j)}, y(j))|j = 1, ..., n\}$ of Section V. Specifically, for the simulations, η was initialized at 0.9 while the two W_{ik} and W_{kt} matrices were randomly generated for the weight parameters of Fig. 1. The number of iterations was set at 1000. After the termination of the algorithm, the weight matrices W_{ik}, W_{kt} that yield the minimum error were chosen for verifying the solution. Amongst the n patterns generated, only the 10% was presented to the neural network while all n patterns were used for validation purposes. Note that in total, $n = 26,931$ patterns were generated from which 97% yield $y(j) = 1$. Let M be the set of all the misclassified patterns and $\boldsymbol{x(*)}$ the pattern for which we want to identify the output value $y(*)$. According to the results, $P(\boldsymbol{x(*)} \in M|y(*) = 0) = 0.06$ (47 out of the 778 patterns with $y(*) = 0$ were misclassified), $P(\boldsymbol{x(*)} \in M|y(*) = 1) = 0.06$ (1568 out of the $26,144$ patterns with $y(*) = 1$ were misclassified), and $P(\boldsymbol{x(*)} \in M) = 0.06$ (1615 out of the total $26,931$ patterns were misclassified). Thus, the proposed solution has an accuracy of 94% while also achieving high accuracy for both levels in set $\boldsymbol{y}$.

Note that the BP algorithm converged to a solution after almost 30min in our MATLAB machine with a CPU@2.60 GHz and 8 GB RAM. After the training procedure (convergence), QoT decisions were made in milliseconds for each connection in the validation dataset.

B. Results for Network B

Network B, is given in detail in Table II (note that Network A could not be given in detail as it consists of a large number of nodes and links that is not easy to illustrate in a research paper). For generating the dataset $\mathcal{D} = \{(\boldsymbol{x(j)}, y(j))|j = 1, ..., n\}$, 1000 requests were generated with the multicast group sizes varying between 1 and 7, thus accounting for both multicast and unicast connections. The same procedure, as the one described in Section V, was followed for generating the dataset for Network B. The Q-threshold was again set at 9dB corresponding to a BER of 10^{-15} and the multicast capable architecture with fixed TXs/RXs described in [2], [9] was again utilized.

In total, $n = 27,439$ patterns were generated and 74% of the n patterns yield $y(j) = 1$. For neural network training, η was initialized at 0.9, while the W_{ik} and W_{kt} matrices were randomly initialized for the weight parameters of Fig. 1. The number of iterations was set at 1000. After the termination of the BP algorithm, the solution that returned the minimum error was chosen. Again, the algorithm converged after almost 30min. The specific weight matrices W_{ik}, W_{kt} that yield the minimum error are given in (9) and (10) respectively.

$$W_{ik} = \begin{bmatrix} -4.8 & 2.84 & -2.66 \\ 8.78 & -9.26 & 8.09 \\ 5.93 & -2.85 & -6.38 \\ 1.97 & -1.78 & -1.02 \\ -5.00 & 5.08 & -4.44 \end{bmatrix} \tag{9}$$

$$W_{kt}^T = [3.67, -5.63, 3.19, -3.43] \tag{10}$$

Amongst the n patterns generated, only the 10% was presented to the neural network, while all n patterns were used for validation purposes. If M is the set of all the misclassified patterns and $\boldsymbol{x(*)}$ is the pattern for which we want to find the associated $y(*)$ value, according to the results, $P(\boldsymbol{x(*)} \in M | y(*) = 0) = 0.0001$, $P(\boldsymbol{x(*)} \in M | y(*) = 1) = 0.12$, and $P(\boldsymbol{x(*)} \in M) = 0.09$. Thus, the proposed solution has an accuracy of 91% while also achieving high accuracy for both levels in set $\boldsymbol{y}$. It is interesting to note that the probability of accepting a call with unacceptable Q-factor is almost zero (99.99%).

TABLE II
NETWORK B

Link	Distance (km)	Link	Distance (km)
$(1,2)$	100	$(1,3)$	100
$(2,3)$	75	$(2,4)$	100
$(1,9)$	80	$(3,6)$	100
$(4,11)$	70	$(4,5)$	60
$(5,6)$	75	$(5,7)$	60
$(6,8)$	100	$(6,13)$	90
$(7,9)$	60	$(9,10)$	60
$(8,10)$	100	$(10,12)$	75
$(11,14)$	100	$(10,14)$	100
$(12,13)$	100	$(13,14)$	60
$(1,4)$	100	$(4,10)$	60
$(2,9)$	70	$(5,11)$	90
$(3,14)$	30	$(13,8)$	40
$(3,5)$	50	$(9,14)$	25
$(6,14)$	50	$(9,12)$	40
$(10,6)$	25	$(7,11)$	40
$(3,12)$	30	$(6,9)$	35
$(13,4)$	40	$(7,1)$	60
$(12,5)$	100	$(3,10)$	90
$(11,12)$	80	$(8,9)$	100
$(11,1)$	100	$(10,11)$	60
$(4,14)$	20	$(5,14)$	100
$(2,12)$	30	$(3,13)$	20
$(5,10)$	40	$(5,9)$	30
$(10,13)$	30	$(9,3)$	100

VII. Conclusions

In this work, a data-driven technique is utilized for analyzing QoT data of previously established connections in a multicast-capable optical network aiming at accurately deciding the QoT of newly arriving connections. A Q-factor formula which is a function of each impairment present is first utilized for generating a QoT dataset. This dataset was analyzed and represented in a way that is independent from the PLIs, but reflects the QoT of each connection requesting to be established in the network. Then, the dataset is used for training and validating the neural network model. The neural network was trained via a gradient decent algorithm. The data-driven QoT model was validated for two network topologies and exhibited high accuracy in both cases. The advantages of the proposed approach, over the existing Q-factor model, is that it is self-adaptive, it is fast, it does not required special measurement equipment, or the existence of a system with extensive processing and storage capabilities. It can therefore be utilized instead of the Q-factor model that lacks self-adaptiveness and requires lab experiments for its re-evaluation upon network changes. It is important to note that the proposed technique can be applied for any network scenario (e.g., flex-grid optical networks, different network architecture/engineering, different modulation formats, etc.), provided that the data selected for the adaptive QoT model are chosen and represented in such a way that is independent from the PLIs but reflects the QoT of the connections requesting to be established in the network of interest.

Acknowledgment

This work was supported by the Cyprus Research Promotion Foundation under Grant TPE/EPIKOI/0311(BIE)/11.

References

[1] C. Politi, et. al, "Physical Layer Impairment Aware Routing Algorithms Based on Analytically Calculated Q-factor," *Proc. OFC*, 2006.

[2] G. Ellinas, et. al, "Multicast Routing Algorithms Based on Q-factor Physical Layer Constraints in Metro Networks," *IEEE Photonics Tech. Letters*, 21(6):365–367, 2009.

[3] N. Antoniades, et. al, "Performance Engineering and Topological Design of Metro WDM Optical Networks using Computer Simulation", *IEEE J. Sel. Areas Commun.,* 20(1):149–165, 2002.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, New York, Inc., Secaucus, NJ, USA, 2006.

[5] J. Oliveira, et. al, "Towards Software Defined Autonomic Terabit Optical Networks," *Proc. OFC*, March 2014.

[6] T. Jimenez, et. al, "A Cognitive System for Fast Quality of Transmission Estimation in Core Optical Networks," *Proc. OFC*, March 2012.

[7] T. Jimenez, et. al, "A Cognitive Quality of Transmission Estimator for Core Optical Networks," *J. Lightw. Technol.*, 31(6):942–951, 2013.

[8] P. Poggiolini, "The GN Model of Non-linear Propagation in Uncompensated Coherent Optical Systems," *J. Lightw. Technol.*, 30(24):3857–3879, 2012.

[9] T. Panayiotou, et. al, "Impairment-aware Multicast Session Provisioning in Metro Optical Networks," *Computer Networks*, 91:675–688, 2015.

[10] R. Rojas, *Neural Networks*, Springer-Verlag, Berlin, 1996.

[11] D. E. Rumelhart, et. al., "Learning Representations by Back-propagating Errors," *Letters to Nature*, 323:533–536, Oct. 1986.

[12] T. P. Vogl, et. al., "Accelerating the Convergence of the Back-Propagation Method," *Biological Cybernetics*, 59(4-5):257–263, 1988.

[13] L. Sahasrabuddhe, et. al., "Multicast Routing Algorithms and Protocols: A Tutorial," *IEEE Network*, 14(1):90–102, 2000.