Evaluating Machine Learning Models for QoT Estimation

Rui Manuel Morais[1,2], **and João Pedro**[1,3]
[1]*Coriant Portugal, Rua Irmãos Siemens 1-1A, 2720-093 Amadora, Portugal.*
[2]*Instituto de Telecomunicações, 3810-193, Aveiro, Portugal.*
[3]*Instituto Superior Técnico, Instituto de Telecomunicações, 1049-001, Lisboa, Portugal.*
e-mail: rui.m.morais@coriant.com, joao.pedro@coriant.com

ABSTRACT
This work evaluates the effectiveness of various machine learning (ML) models when used to predict the Quality of Transmission (QoT) of an unestablished lightpath, speeding up the process of lightpath provisioning. Three network scenarios to efficiently generate the knowledge database used to train the models are proposed as well as an overview of the most used ML models. The considered models are: K nearest neighbors (KNN), logistic regression, support vector machines (SVM), and artificial neural networks (ANN). Results show that, in general, all ML models are able to correctly predict the QoT of more than 95% of the lightpaths. However, ANN is the model presenting better generalization, correctly predicting the QoT of up to 99.9% of of the lightpaths.
Keywords: Quality of Transmission, machine learning, transport networks, software defined networking.

1. Introduction

The ultimate goal for telecom operators is twofold: get the most from installed equipment by operating more closely to optimality in order to reduce Capital Expenditures (CapEx); and automatize network provisioning, reconfiguration, and healing in order to reduce Operational Expenditures (OpEx). Nowadays, Software Defined Networking (SDN) is being employed to achieve these goals. However, today's network is rapidly growing in terms of number of connected devices, which increases the complexity of network management to a level that becomes very challenging even with SDN [1]. In this environment, Knowledge Defined Networking (KDN) is arising as a candidate to implement a self-driving network [1]. KDN is a complementary solution for SDN that introduces reasoning processes and ML techniques into the control plane of the network with the aim of enabling an autonomous and rapid operation. Nowadays, monitoring is already extensively used in optical networks and nodes start to be equipped with computing and storage capabilities, allowing them to have a rich view of the network analytics. Moreover, SDN and flexible hardware are now state-of-the-art technologies, enabling the network to be remotely configured and adapt to changing conditions. Therefore, the missing piece to pave the way to KDN is the introduction of ML into the control plane of the network.

One of the key problems in the establishment of a self-driving optical network is the automatic provisioning of lightpaths. Lightpahts accumulate impairments that may degrade the QoT of the signal to a degree that avoids its correct detection at the receiver end. Thus, to quickly provision a new lightpath, accurate and fast QoT evaluation is required. However, assessing the optical signal quality involves complex and time consuming computations [2]. As a consequence, strategies for accurate online QoT evaluation should rely on a compromise between accuracy, computation time, storage capacity, and actuality of network state information. Recently, ML is being proposed for lightpath QoT estimation [3]–[5]. ML are models that can be trained with known examples of the problem, achieving a generalization that allows to predict unknown events. For instance, ML could utilize data from already established lightpaths to predict the QoT of unestablished ones. However, ML models suffer from what is usually called "*No free lunch*" theorem, i.e. no single model works best for every problem [6]. As a result, it is paramount to evaluate and compare different ML models for QoT estimation in order to gain insight on the most promising ones for this specific application.

This work describes three network scenarios to generate meaningful optical performance data, which are used to train ML models, and compares various models for QoT estimation.

2. Network Scenario and Knowledge Acquisition

To have a robust and reliable ML model, data from the network itself, namely from long term network monitoring, is mandatory. Thus, Fig. 1 presents three workflows to acquire data for training ML models for QoT estimation. In the first option (see Fig. 1(a)), the optical network is working on regular operation, i.e. requests for lightpaths arrive to the SDN controller on-demand. In order to fulfill the request the SDN controller asks the optical performance model (OPM) to estimate its QoT. The OPM, using accurate network parameterizations from the monitoring system sends to the SDN controller the acceptance or rejection of the request. At the same time, this information is saved in the knowledge database. Depending on the network lightpath request rate, this option can have a long ramp up phase due to the number of examples that ML models require to achieve a trustful generalization. To contour this limitation, an alternative is the use of dummy requests (see Fig. 1(a)). Exploiting this option, the SDN controller randomly generates dummy requests by itself. Then, the OPM estimates the QoT of these requests and sends the information to the knowledge database. In this way the ramp up phase can be faster as more data are available for training. Moreover, the dummy requests can include very diverse paths (e.g., very long ones), many of which would probably never be generated during normal operation. Thus, this option not

only accelerates the ramp up phase but also improves the generalization of the ML. The two frameworks proposed before rely on an OPM. In order to avoid this requirement, a third option can be employed. More precisely, the optical network establishes lightpaths on request and their QoT is accessed only through the monitoring system (see Fig. 1(b)). The result of the monitoring system is then fed to the knowledge database and to the SDN controller. The drawback of this option is that only after the lightpath is established we can obtain the monitoring data.

After the aforementioned procedures, the ML model can be trained with the available data and then be used in online operation. Figures 1(c) and (d) present the worflows with and without an OPM, respectively. The optical network performs a request to the SDN controller. If an OPM is available, the SDN controller asks the trained ML model and also to the OPM to validate the lightpath. Whenever the prediction is greater than a given threshold we rely on the ML answer. However, if the prediction is smaller than the given threshold a second opinion is asked to the OPM or to the monitoring system. Either way, after the request is processed the new data is fed to the knowledge database and the ML model is re-trained with this new information. As time goes by this will allow to decrease the threshold up to a level such that the SDN controller can rely on the ML model for predicting the QoT of almost all lightpaths requests.

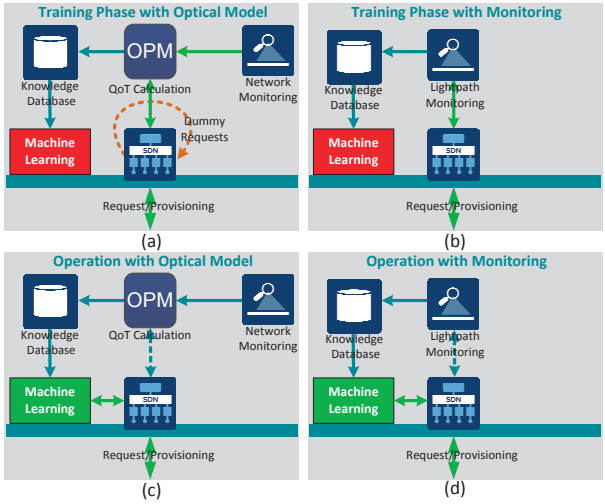


Figure 1. Workflows to acquire data from an optical network and generate a knowledge dataset.

3. Machine Learning Models

A ML model is presented with examples and their respective outputs. The model is then considered successfully trained when the difference between its predicted outputs and the outputs given in the training set is satisfactory. It is then expected that the trained ML model can generalize well, enabling the prediction of examples that were not explicitly used in the training phase. Each example, s_e, is composed by m features, x, and an output y_e, $s_e = (x_e^1, \ldots, x_e^m, y_e)$. The generalized model becomes a function $f(x_e^1, \ldots, x_e^m) = \hat{y}_e$ that minimizes the difference between the predicted, $\hat{y}_e$, and the real, y_e, target.

3.1 K Nearest Neighbors

The KNN technique is a non parametric lazy training algorithm [7]. It is non parametric as it does not make any assumptions on the underlying data distribution. This can be a useful property since most real data does not obey the typical theoretical assumptions made (e.g. gaussian process, linearly separable, etc.). It is also a lazy algorithm because it does not use the training data points to do any generalization. The training phase of the algorithm consists only on storing the data of the training examples. The basic idea of the KNN algorithm is that a point closer to another point will likely have the same output. Thus, in the prediction phase the classification of the query example is performed by a majority voting of its K nearest neighbors, being assigned the most common output. In order to determine the distance between two multidimensional points, various metrics can be used. Here, we evaluate two metrics: the Euclidian, and the Cosine [7]. One of the drawbacks of the KNN algorithm is that when the data is skewed, i.e. one of the classes is much more frequent, the prediction can be biased [7]. One way to mitigate this problem is to weight the classification, i.e. the weight to the K nearest neighbors is proportional to the inverse of the squared distance of the neighbor to the query object [7].

3.2 Logistic Regression

The logistic regression is a parametric model that is based on the properties of the logistic function. The logistic function can take any real input but the output always takes values between zero and one [7]. Thus, it can be interpreted as a probability and if the output is smaller than 0.5 the example is classified as false (zero) whereas if it is higher the example is classified as true (one). The goal of the logistic regression is then to find the logistic function parameters such that the difference between the predicted, $\hat{y}_e$, and the real, y_e, output is minimized. After performing the optimization procedure new predictions to query examples are easily achieved using the logistic function.

3.3 Support Vector Machines

SVMs are ML algorithms that construct a hyperplane or set of hyperplanes in a high- or infinite-dimensional space [7]. In linear SVM the goal is to find a linear hyperplane that separates all data points of one class from those of the other class, i.e. determine the hyperplane that has the largest margin between the two classes. In spite of linear SVM being able to find the best hyperplane that separates the two classes, it often happens that the sets are not linearly separable in that space. However, utilizing a technique known as the kernel trick, SVM

can become much more flexible and support various types of nonlinear decision boundaries [7]. The kernel trick performs a mapping of the original feature space to a higher-dimensional space, in which the separation between the groups is clear, or at least clearer. In this work, we consider linear SVM and the quadratic, cubic and the gaussian (or radial basis) kernels with $\gamma = 1$ and $\gamma = 3$ [7].

3.4 Artificial Neural Networks

An ANN consists of an interconnected group of artificial neurons that have the ability to be trained from empirical data. The neuron is the basis for designing ANN and is composed by three basic components: a set of synapses, each of which characterized by a weight, an adder for summing the input signals, and an activate function to control the amplitude of the output [7]. The model also includes an externally applied bias that has the effect of increasing or decreasing the input of the activation function. ANNs usually make use of more than one neuron, thus the neurons are organized into layers to form a network. The input layer is simply composed by the values of the examples in the training set, constituting the inputs to the next layer of neurons. The next layer is called a hidden layer. An ANN may contain one or several hidden layers, each one formed by one or more neurons. The final layer is the output layer, where the estimated output is returned. The training process in ANN occurs within each cycle or epoch, i.e. each time the network is presented with a new example. After all weights being initialized with random values, the ANN estimates an output for a given example s_e and current weights θ, $\hat{y}_e(\theta)$. If the training process is not yet completed, $\hat{y}_e(\theta)$ will differ from the target y_e. The goal is then to minimize an error function that measures the difference between the predicted and observed outputs.

4. Data Generation and Features Selection

In order to evaluate the robustness of the ML models, and due to the lack of real network monitoring data available to perform meaningful comparisons, we have generated synthetic data for 3 reference network topologies. The physical characterization of these networks is presented in Table I.

TABLE I. Network Characteristics.

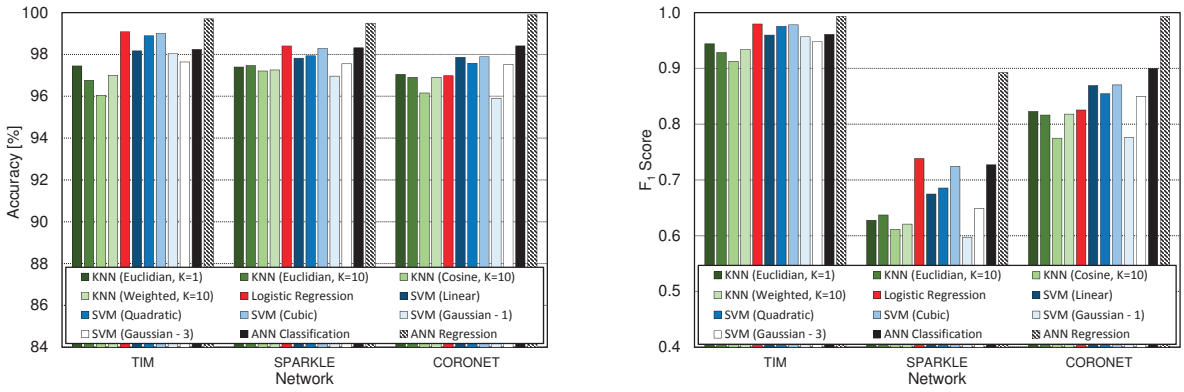
Network	Nodes	Links	Av. Link Length [km]	Fiber Types	Number of Paths
TIM [8]	44	71	174.3	SSMF LEAF	1134272
SPARKLE [9]	49	72	388.0	SSMF LEAF	241115
CORONET [10]	75	82	396.0	SSMF	105320

For each network we calculated all paths below 4600 km and upper bounded to 1200 paths per node pair. Due to memory limitations, for the CORONET network the number of paths was bounded to 400 per node pair. Afterwards, the determination of the residual margin to infer the feasibility of the lightpath is assessed using the Gaussian noise (GN) approach for nonlinear interference (NLI) caused by Kerr effect [11] with extra filtering and equipment related penalties [8]. Five different optical channel formats were considered, assuming a symbol rate of 32 Gbaud and five modulation formats: QPSK, 8QAM, 16QAM, 32QAM, and 64QAM. The detailed description of the optical performance model used as well as the required OSNR in B2B configuration for each of the modulation formats, and the fiber types characterization can be found in [8]. Each example, i.e. each lightpath, is composed by the following set of 13 features: Number of hops; Number of spans; Total length [km]; Average link length [km]; Maximum link length [km]; Link lengths standard deviation; Average span attenuation [dB]; Average dispersion [ps/nm/km]; and Modulation format. The target output is a binary response: feasible / unfeasible.

5. Results and Discussion

The 12 ML models evaluated and compared are: KNN, Euclidian with $K = 1$; KNN, Euclidian with $K = 10$; KNN, Cosine with $K = 10$; KNN, Euclidian Weighted with $K = 10$; Logistic Regression; Linear SVM; Quadratic SVM; Cubic SVM; Gaussian SVM with $\gamma = 1$; Gaussian SVM with $\gamma = 3$; ANN for classification; and ANN for regression. The training set used corresponds to 0.1% of the total data generated. Therefore, 5 318 examples were used for TIM, 1 161 for SPARKLE, and 530 for CORONET. The accuracy of the different ML models is then calculated considering predictions over the complete generated set, i.e. 5 671 360 examples for TIM, 1 205 575 for SPARKLE, and 526 600 for CORONET. Note that accuracy is defined as the ratio between the correctly predicted examples and the total number of examples. Accuracy is the most common comparison metric for ML models, however it can also be misleading in case positive and negative examples are not evenly distributed. Therefore, in addition to accuracy we also compared the F_1 score obtained [7]. An F_1 score equal to 1 corresponds to a perfect match, i.e. all predictions are correct.

Figure 2 displays the obtained results for accuracy (see Fig. 2(a)) and F_1 score (see Fig. 2(b)). By simple inspection of Fig. 2(a), it can be observed that all evaluated models show an accuracy higher than 95%. These



(a) Accuracy (b) F_1 Score

Figure 2. (a) Accuracy and (b) F_1 score obtained considering all examples for models trained with 0.1% of the generated data.

results confirm the usability of ML models for lightpath validation. Regarding the comparison of the models, ANN tends to have higher accuracy, whereas KNN shows the worst (see black and shades of green bars in Fig. 2(a)). SVM models accuracy tends to rely between ANN and KNN (see shades of blue bars in Fig. 2(a)). The logistic regression (see red bars in Fig. 2(a)) presents a good trade-off between accuracy and training complexity, being the fastest model among all and tending to show higher accuracy than KNN and even SVM in some cases. Regarding F_1 score, as can be seen in Fig. 2(b), the SPARKLE network obtained the lowest F_1 score. This is mainly explained by the fact that in the SPARKLE network almost 98% of the examples are negative examples. The ANN for regression is the model with higher F_1 score for all networks. Even in the SPARKLE network, the ANN for regression presents a F_1 score of 0.89. Regarding the KNN models, it can be seen that KNN Euclidian with $K = 1$ tends to present the higher F_1 score and accuracy among all KNN parameterizations. The unique exception is in the SPARKLE network, where the KNN Euclidian with $K = 10$ outperforms the former. Fixing the Euclidian metric, $K = 1$ always obtains better results than $K = 10$, even when weighted metric is being used. Moreover, using $K = 10$, the Euclidian weighted metric presents the best results. Using more than one neighbor to perform classification showed to penalize the classification. Regarding the various SVM parameterizations, it can be seen in Fig. 2 that the cubic SVM presents higher accuracy and F_1 score for all networks (see lighter shade of blue bars in Fig. 2). Moreover, fixing the Gaussian kernel, $\gamma = 3$ tends to outperform $\gamma = 1$, being the unique exception the SPARKLE network. In spite of the good results presented by KNN and SVM models, ANN for classification and logistic regression are the models presenting the second best results. For simpler networks, i.e. employing just one type of fiber and with fiber spans of almost equal length, ANN for classification obtains the second best result (see black bars of CORONET in Fig. 2), whereas for networks with a more complex fiber mix logistic regression obtained the second position (see red bars of TIM and SPARKLE in Fig. 2). Logistic regression focuses on maximizing the probability of data classification, in opposition to SVM, which focuses on maximizing the distance of the closest points to the margin. Therefore, in more complex problems it is easier to find an accurate probability of classification than to explicitly find a decision boundary.

6. Conclusions

This work has proposed three frameworks to acquire meaningful data for a knowledge database that can be used to train ML models for lightpath validation in a SDN-controlled optical network. It has also overviewed and compared various ML models to evaluate the feasibility of a lightpath. The models were: KNN, logistic regression, SVM, and ANN. Results show that ML is a valid alternative to estimate the QoT of a lightpath. Noteworthy, ANN proved to be the model achieving the best generalization with accuracies in the order of 99%.

Acknowledgments

This work was partially funded by FCT/MEC through national funds and by FEDER - PT2020 partnership agreement under the project UID/EEA/50008/2013 (action SoftTransceiver) and by the H2020 Metro-Haul project under grant agreement number 761727.

References

[1] A. Mestres, *et al.*, *ACM SIGCOMM CCR*, 47(3), 2017.
[2] I. Sartzetakis, *et al.*, *JOCN*, 8(9), p. 676-688, 2016.
[3] R. Borkowski *et al.*, *JOCN*, 7(2), p. A344-A355, 2015.
[4] T. Jimenez, *et al.*, *JLT*, 31(6), p. 942-951, 2013.
[5] L. Barletta, *et al.*, *OFC2017*, p. Th1J.1, 2017.
[6] D. Wolpert, *Neural Computation*, p. 1341-1390, 1996.
[7] S. Shwartz and S. David, Cambridge University Press, 2014.
[8] J. Pedro, *JOCN*, 9(4), p. C35-C44, 2017.
[9] A. Eira, *et al.*, *JOCN*, 8(7), p. A101-A115, 2016.
[10] J. Pedro, *JOCN*, 7(2), p. A190-A199, 2015.
[11] P. Poggiolini, *JLT*, 30(24), p. 3857-3879, 2012.