Predictive Autonomic Transmission for Low-Cost Low-Margin Metro Optical Networks

Marc Ruiz[1], Fabien Boitier[2], Behnam Shariati[1], Patricia Layec[2], and Luis Velasco[1*]
[1] Optical Communications Group (GCO), Universitat Politècnica de Catalunya (UPC), Barcelona, Spain.
[2] Nokia Bell Labs, Nozay, France

*Corresponding author: lvelasco@ac.upc.edu

Abstract— Low-cost low-margin implementation plays an essential role in upgrading optical metro networks required for future 5G ecosystem. In this regard, low-resolution Analog-to-Digital Converters (ADCs) can be used in coherent optical transponders to reduce cost and power consumption. However, the resulting transmission systems become more sensitive to physical layer fluctuations like the events caused by fiber stressing. Such fluctuations might have a strong impact on the Quality-of-Transmission (QoT) of the signals. To guarantee robust operation, soft-decision Forward Error Correction (FEC) techniques are required to guarantee zero post-FEC Bit Error Rate (BER) transmission, which could increase the power consumption of the receiver and thus operational expenses. In this paper, we aim at minimizing power consumption while keeping zero post-FEC errors by means of a predictive Autonomic Transmission Agent (ATA) based on Machine Learning (ML). We present a sophisticated ATA model that, taking advantage of real-time monitoring of state of polarization (SOP) traces and the corresponding pre-FEC BER, predicts the right FEC configuration for short-term operation, thus requiring minimum power consumption. In addition, we propose a complementary long-term prediction of excessive pre-FEC BER to enable remote reconfiguration at the transmitter side through the network controller. A set of experimental measurements is used to train and validate the proposed ATA system. Exhaustive numerical analysis allows concluding that ATA based on Artificial Neural Networks (ANN) predictors achieve the maximum QoT robustness with 80% power consumption reductions compared to static FEC configuration.

***Keywords*: Autonomic Transmission, BER prediction, Network Automation.**

1 INTRODUCTION

Given the network traffic and services sustained growth forecast for the coming years [1], optical networks, specifically those in the metro segment, need to be redesigned. In this regard, low-margin operation of optical networks with reduced cost is becoming attractive for operators dealing with such traffic increment [2]; low-margin operation requires both advanced optical transmission technologies and real-time solutions for network performance estimation.

The introduction of the coherent transmission technology in metro networks [3] requires the redesign of classical long-haul systems aiming at minimizing capital and operational (mainly from power consumption) costs [4]. A solution to reduce the costs of coherent optical transponders is to reduce the number of bits of Analog-to-Digital Converters (ADCs), from 8 bits used in long-haul transmission to just 4-5 bits (*low-resolution* ADCs). However, this causes a penalty in terms of performance and thus, in the resulting Bit Error Rate (BER) before the Forward Error Correction (FEC) [5].

To keep maximum Quality of Transmission (QoT) performance, *soft-decision* FEC techniques, such as iterative Low-Density Parity Check (LDPC) decoding [6] can be used, where several iterations are performed. However, the number of LDPC iterations has a strong impact on the power consumption; therefore, adapting the number of LDPC iterations can be considered to save power. In fact, according to [6], up to 82% power savings could be potentially achieved by adjusting the number of LDPC iterations according to the connection's length; note that the longer the connection, the higher the number of iterations. Specifically, connections shorter than 1,200 km need no additional LDPC iterations, which opens the possibility to reach maximum power savings in metro networks where connections' length are shorter. In addition, the number of LDPC iterations can be dynamically determined based on the estimated quality of signal, as proposed in [7] for wireless communications.

Reducing the-resolution of ADCs causes a penalty in terms of performance of transmission systems [5], and thus, the resulting pre-FEC BER is more sensitive to physical fluctuations, such as the state of polarization (SOP) rotation. Hence, solutions for dynamic LDPC configuration based on QoT estimation need to consider SOP fluctuations, which can be caused by external events like hitting, bending, or shaking. In fact, fluctuations causing fast variation on the polarization state can drastically affect system margins planned by the operator [8]. In low-margin networks, the impact of soft-failures (i.e., fiber bending) that could lead to hard failures (fiber cut) increases [9], [10]. Note that pre-FEC BER might be noticeably impacted by SOP fluctuations [11], and therefore, the number of LDPC iterations needs to be configured to a conservative high value to ensure robust transmission at the expense of higher power consumption.

The solution, in view of the infrequent nature of such events, is to work toward intelligent approaches able to adapt the FEC decoder configuration to the actual needs, aiming to reduce power consumption without compromising QoT. To that end, Machine Learning (ML) techniques [12], need to be placed as closer to the sources of monitoring to reduce time for data analysis and decision making [13]; specifically, they can be applied to build autonomic transmission systems able to self-adapt their configuration in response to changing conditions. *Automating* the optical layer is a key enabler for the sustainable growth of the networks, due in part to the increasing dynamicity and complexity of optical transport networks.

In this regard, ML-based algorithms need to focus on the anticipated detection of those events (e.g., SOP fluctuations) that can potentially produce bursts of errors to enable decision making, resulting in the reconfiguration of the receiver or the transmitter. Specifically, the reconfiguration of the transponders is enabled by the extensive work carried out in the last years to design architectures based on Software Defined Networking (SDN) principles for the operation optical networks [14], complemented with monitoring and data analytics capabilities targeting *autonomic networking* [15]-[18]. Although these architectures have shown their feasibility to implement autonomic networking in disaggregated scenarios [19], the possibility of implementing *autonomic transmission* by running self-configuration algorithms inside the transponders will enable a fine grain control of the transmission systems. Moreover, autonomic transmission allows keeping a dynamic and smart trade-off between QoT assurance and cost reduction, e.g., operating at a target performance level with minimum cost.

In line with the above, in our previous work [20], we presented an *Autonomic Transmission Agent* (ATA) running inside of coherent transponders, focused on anticipating the detection of pre-FEC BER degradation of individual lightpaths by monitoring real time the Stokes parameters representing the SOP. Two use cases were introduced, namely: *i*) reconfiguration of the local receiver based on BER degradation prediction, and *ii*) reconfiguration of the remote transmitter by changing transmission parameters such as the FEC coding rate, modulation format or baud-rate. Note that the requirements of the anticipated detection (or prediction) are different for the local and the remote node; whereas local reconfiguration can be based on short-term prediction (next few ms.), remote reconfiguration requires message exchange through the SDN controller, which entails additional time and thus, longer term prediction (next tens of ms.)

In this paper, we extend our previous work by closing the loop to tune the number of LDPC iterations according to the evolution of SOP and pre-FEC BER. Specifically, the contribution of this paper is two-fold:

- the details of the ATA are presented in Section II. Description of the processes and modules within ATA are detailed from a generic autonomic transmission perspective. The generic ATA modules are adapted to a specific use case where the number of LDPC iterations is configured dynamically to guarantee zero post-FEC errors, while minimizing power consumption.
- the ML models used in ATA prediction engines are described in Section III. In particular, we improve the Naïve-Bayes (NB) approach presented in [20] and present a model based on Artificial Neural Networks (ANN) as alternative methodology.

The discussion is supported by numerical results based on experimental measurements targeting at validating the practical feasibility of ATA, as well as quantifying the expected benefits of the selected use case.

2 PREDICTIVE AUTONOMIC TRANSMISSION

A. ATA Architecture

In this section, we propose to augment optical transponders with autonomic capabilities by including the ATA detailed in Fig. 1. Considering a point-to-point connection, the ATA receives SOP and pre-FEC BER monitoring data from the receiver (labelled 1 in Fig. 1) and analyzes that data. Based on that analysis, the ATA provides guidelines and operational decisions for the local receiver (control loop 1-L) or the remote transmitter through the control plane (control loop 1-R).

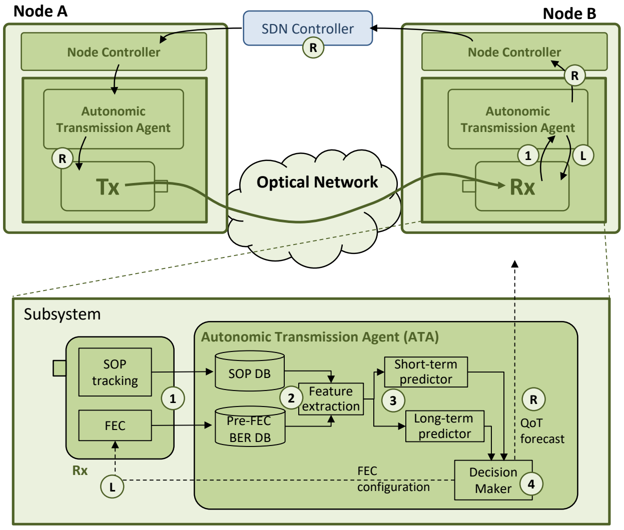

Fig. 1. Autonomic transmission and detail of receiver side.

Additionally, Fig. 1 illustrates how ATA works at the subsystem level for both local and remote reconfiguration. Let us assume that the transponder is able to continuously collect the Stokes parameters tracking the SOP and the pre-FEC BER (1) monitoring data at the receiver side. Since these measurements can be collected at a fast rate, monitoring data is conveniently summarized to reduce the amount of data to be stored in the local databases (DB) while keeping fine enough granularity. Note that all data aggregation and analysis procedures need to be designed to allow real-time operation avoiding any computational bottleneck that could make unpractical the overall ATA operation.

Every time a new aggregated measurement is stored, a three-phase data analysis process is executed. First, the *feature extraction module* transforms a vector with the most recent stored input data (according to a pre-configured window size) into a number of selected features describing the current state and the observed evolution of both the SOP and pre-FEC BER (2). Extracted features feed two prediction modules (3): the *short-term predictor* forecasts the proper FEC decoder configuration required to guarantee robust transmission in the next short-term time interval, whereas the *long-term predictor* is designed to provide a forecast of the expected QoT evolution for a longer time interval. Finally, decision making process (4) analyzes the outputs of predictors and decides whether a local (L) or a remote (R) reconfiguration needs to be performed.

The decision-making process plays the important role of guaranteeing local system robustness in terms of performance; even under the assumption that short and long-term predictors are reliable and provide accurate predictions for conditions similar to those of training environments, the system could be harmed if an unforeseen situation occurs. In that case, the decision maker needs to react quickly and guarantee transmission performance over any other objective, e.g., by configuring FEC decoder for the most conservative situation.

The next section presents a specific use case providing further details on scenario identification and both short and long-term predictors.

B. Use Case: dynamic configuration of LDPC iterations

Let us consider a use case where ATA is used to dynamically configure the number of soft-decision FEC decoding iterations according to the forecast needs. The objective of this dynamic configuration is to fit the smallest number of iterations that results in zero post-FEC errors. Recall from the introduction that reducing the number of iterations potentially leads to power consumption savings. Prediction of that minimum number of iterations is performed by means of a ML-based procedure receiving as input historic monitoring data streams with the evolution of the Stoke parameters and the corresponding pre-FEC BER, both measured in the receiver.

Fig. 2 presents a comprehensive example aiming at illustrating ATA operation for local receiver reconfiguration in response to: *i*) a strong SOP fluctuation and *ii*) a gradual pre-FEC BER degradation. From top to bottom, four graphs represent the evolution with time of four relevant metrics for a given optical connection: *i*) Stoke parameters; *ii*) pre-FEC BER; *iii*) the number of LDPC iterations configured in the receiver in the range [*minIte, maxIte*] ([2,8] in the example); and *iv*) the expected relative power consumption savings compared to statically configuring *maxIte* iterations to guarantee transmission performance under any scenario.

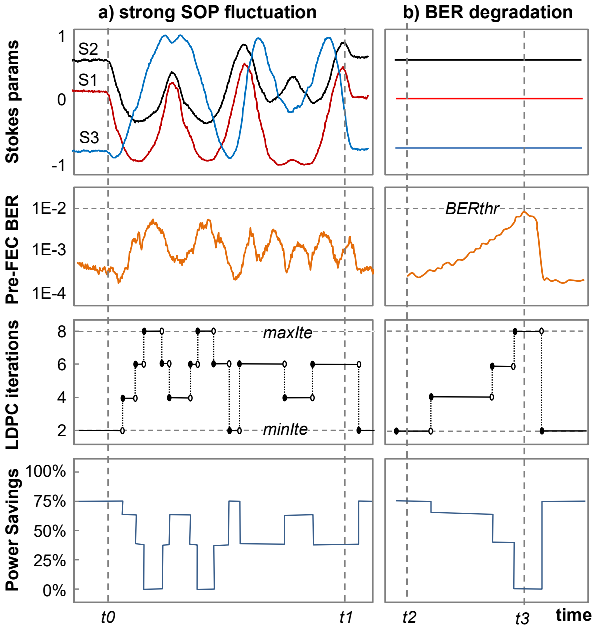

Fig. 2. Illustrative example.

Fig. 2a shows the first scenario where an event starting at t_0 causes a strong SOP fluctuation. Before t_0, pre-FEC BER magnitude allows error-free post-FEC transmission by configuring the number of LDPC iterations to *minIte*; this is the number of iterations that this specific connection requires due to its inherent physical characteristics, such as length [6]. As a result of the reduced number of required iterations, large power consumption savings (up to 75%) can be observed with respect to the conservative static FEC configuration. After the event in t_0, SOP fluctuations impact pre-FEC BER and the number of LDPC iterations are tuned according to the short-term predictor. Note that during the event, power consumption savings drop in benefit of QoT assurance. Once SOP fluctuation stabilizes in *t1*, *minIte* can be restored and consequently power consumption savings are maximized.

In the second scenario in Fig. 2b, QoT degradation is due to a soft-failure that produces pre-FEC BER increment (see [21], [22]). Note that, under the presence of such soft-failure, the autonomic transmission subsystem guarantees QoT by adjusting LDPC iterations based on the short-term prediction. However, reaching some BER threshold (*BERthr*) could make that zero post-FEC error cannot be achieved and the connection will be consequently torn down. In order to avoid this, long-term prediction plays an important role by anticipating such violation with enough time so that a notification can be sent to the SDN controller. The SDN controller can reconfigure the connection by, e.g., rerouting it avoiding a potentially failed component. The illustrative example clearly shows the crucial role of the decision maker module to ensure the proper operation of the proposed ATA.

3 ATA OPERATION

In this section, we concentrate on the first scenario described in the previous section, where a strong SOP fluctuation produces high pre-FEC BER. For such scenario, the short-term predictor outputs the optimal number of LDPC iterations; note that the range of LDPC iterations is typically limited, e.g., even numbers between 0 and 10 [6]. As for the long-term prediction, the output can be the pre-FEC BER forecast categorized into labels describing the expected behavior (*belowThr, aboveThr*). Therefore, we consider implementing both short-term and long-term predictors as classifiers [12]. In the rest of this section, we first focus on data preparation and feature computation for model training, validation, and utilization. Then, two different approaches are presented to model classifiers: Naïve-Bayes (NB) and Artificial Neural Networks (ANN). The pros and cons of each approach are highlighted. Finally, the general operation of the decision-making module is defined.

A. Notation and feature computation

Fig. 3a plots the evolution of the Stokes parameters and Fig. 3b the pre-FEC BER during a time interval where an event causes fluctuation of both measurements. Monitoring data for both parameters are used as inputs for the predictors; note that each Stoke parameter is treated as an individual variable. In addition, the actual class of the response variable (output) within that interval is shown in Fig. 3c. As already mentioned, the response variable can be either the number of LDPC iterations in the short-term predictor (multiclass classifier) or the expected pre-

FEC BER evolution in the long-term predictor (binary classifier). Let us now suppose that current time is on reference time *t*. Our proposed predictive model receives as input the monitored data within the time window [*t*-*w*+1, *t*] (shaded areas in Fig. 3 highlight the monitoring data stream segments for each input data variable). Data granularity depends on raw samples aggregation, being the number of aggregated samples in the aforementioned window equal to *w*. With such input data variables, the model predicts the expected class value of the response at time *t*+*δ*.

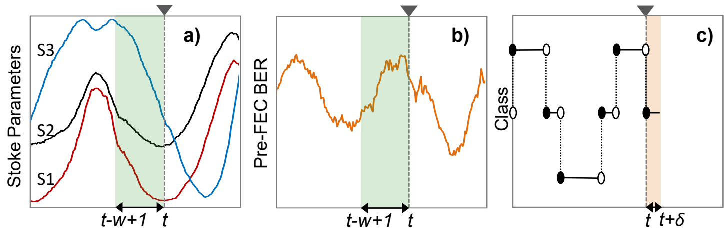

Fig. 3. SOP and pre-FEC BER-based prediction

Aiming to reduce the amount of input data, we summarize each input variable into two relevant features that, for the sake of a broader utilization, are normalized within the range [-1, 1]. Let us denote X_i as the input variable *i* and $x_i(t)$ the value that input variable *i* takes at time *t*. It is worth noting that, within a moderated window length, the evolution of an input variable follows a reasonable linear approach (see Fig. 3 for an illustrative example). Then, data in the window can be approximated with only two relevant features: the current value (*cur*) and the trend (*trend*). The *cur* feature represents the current status of the variable at time *t* and it is computed as:

$$cur_i(t) = 2 \cdot \frac{x_i(t) - \min(X_i)}{\max(X_i) - \min(X_i)} - 1 \tag{1}$$

The *trend* feature approximates the increasing/decreasing observed within the time segment [*t*-*w*, *t*] and it is computed as:

$$trend_i(t|w) = \frac{cur_i(t) - cur_i(t-w+1)}{2} \tag{2}$$

Finally, let us denote *Y* as the response variable. Then, the predictive classifier can be formally stated:

$$Y(t|w) \sim f(cur_i(t), trend_i(t|w) \mid \forall i = \{s_1, s_2, s_3, BER\}) \tag{3}$$

B. Naïve-Bayes model

The NB predictor can be defined as a simple hash table, where each key encodes a specific combination of the input features and contains the response variable(s) value(s). The prediction that the model returns is based on the observations and could vary depending on the use case, e.g., the maximum, the average, a probability distribution, etc. In our case, the response variable is the worst number of LDPC iterations observed for that specific input features combination.

The NB technique provides one of the simplest ways to obtain a predictive model based on observed data. In fact, it allows considering any possible complex correlation between input features and the response without any prior assumption on such correlation and the computational cost of generating predictions almost negligible; this allows for very fast operation. The drawbacks are: *i*) it requires input features taking a discrete finite number of potential values and, in case of continuous features (as in our case), a practical and efficient way to discretize them, and *ii*) only those input features combinations present in the training data are actually learned. Thus, the observation of unknown combinations during operation can lead to inaccuracies potentially making harmful decisions, so special cautions must be taken during the training phase to avoid them.

Aiming at clarifying the issues above for our specific use case, Table I shows the training algorithm for the NB model. It receives a set of training data *D* consisting of a set of training samples, each characterized by a tuple <*F*, *y*>, where *F* is an ordered list of input feature values and *y* the related response variable class. Moreover, it receives the list *R* with all possible values for the response class sorted from best to worst according to the model scope. For instance, for short-term prediction, classes are sorted in increasing number of LDPC iterations, whereas for long-term prediction they are ordered from best to worst, i.e., from *belowThr* to *aboveThr*. Finally, the algorithm receives a parameter *p* that is used to discretize features, as detailed further, keeping control on the size of the model.

Table I. Training algorithm

```
INPUT: D, R, p
OUTPUT: model
 1:  $g \leftarrow \emptyset$
 2:  for <F,y> in D do
 3:      $key \leftarrow 0$
 4:      $i \leftarrow 0$
 5:      for f in F do
 6:          $f_{disc} \leftarrow$ discretize($f$,$p$)
 7:          $key \leftarrow key + f_{disc} \cdot$ p^i
 8:          $i$++
 9:      $g[key]$.push($y$)
10:  for key in g.keys() do
11:      $Y \leftarrow g[key]$.pop()
12:      $y^* \leftarrow$ selectClass($Y$, $R$)
13:      $g[key] \leftarrow y^*$
14:  $g[-1] \leftarrow$ selectWorstClass($R$)
15:  return g
```

After initializing the empty model (line 1 in Table I), all the samples are processed to build an initial model structure (lines 2-9), where a numerical key is computed from the normalized feature values. Specifically, a key is a *p*-base positive integer, where each digit belongs to one of the features that has been properly discretized. Discretization is as simple as converting a real normalized feature in the range [-1, 1] into an integer number between 0 and *p*-1, assuming uniform distribution within the range. Once the numerical key is obtained, the observed response value of the sample is temporarily stored.

Next, a second phase is executed to both reduce the size of the model and facilitate its utilization during operation (lines 10-15). In this phase, the list of different response samples stored for each single numerical label are summarized to only one; to allow robust operation, we selected the worst class among all the classes observed. For example, in the case of modelling the short-term predictor, if five different samples under label '2641416' were observed: ['2', '2', '2', '4', '2'], the selected response will be '4' as it is one entailing more iterations. In addition, the worst among all possible classes, i.e., '10' in our case, is set up as the *null* value that will be returned when a non-trained label is requested (line 14). In this regard, it is worth noticing that the algorithm required to produce a prediction with this model comprises two consecutive steps: 1) computing the feature numerical label as detailed in the training algorithm; and 2) returning the class stored for that label if that label exists; otherwise, it returns the null class.

C. ANN model

The motivation behind the choice of ANNs as candidate ML model for the proposed use case is based on the wide and extended use of this technique for a plethora of use cases and applications [23]. Its flexibility allows configuring different ANN models for solving the same problem, which increases the options of finding a robust and accurate prediction tool. It is also less restrictive than NB, in general, e.g. it allows working with continuous, discrete, and categorical data at the same time. On the contrary, the training procedure is much more complex and could lead to bad results if it is wrongly configured or not adapted to the specific data. As for the utilization of the model in real operation, it has similar storage needs and computational complexity than NB.

Fig. 4a shows the scheme of an ANN implementing the generic model in equation (3). The number of input neurons equals the number of features identified in Section 3.A. On the other hand, one output neuron per each class of the response variable is defined. In this way, weak binary decisions (0/1) per class are independently taken, which are processed to provide a strong multiclass classification [23]. In this regard, the most conservative class, i.e., the highest number of LDPC iterations, is returned if more than one positive binary decision is obtained. Regarding the hidden layer, the number of neurons is defined as a function of the amount of training data samples. To avoid overfitting, we aim at keeping moderated the ratio between model coefficients and training samples, e.g., 1/10. Therefore, the number of hidden neurons will be properly setup to allow achieving such target ratio.

Activation functions configured at hidden and output neurons are *softmax* and *rectified linear units*, respectively. This configuration is often used in classification models due to its good performance in terms of model accuracy and training speeding-up. The selected backpropagation training algorithm is based on mini-batch gradient descent optimization, targeting at minimizing the cross-entropy at each training epoch [24]. The backpropagation algorithm is executed until a two-fold stop criteria is met: either a maximum number of training epochs (10000) is reached or a consecutive number of training epochs (1000) without significant cross-entropy improvement (threshold<0.01) is reached.

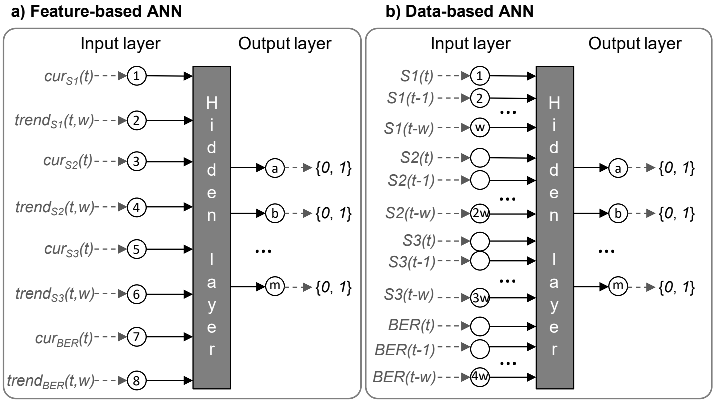

Fig. 4. ANN model variants

As stated previously, ANN is flexible in the sense that various ANN-based designs could be devised. In this regard, an alternative would consist in a "feature-free" approach, where aggregated monitoring data is directly input to provide the desired output (Fig. 4b). The main advantage of this approach is that information loss due to feature computation can be avoided. On the other hand, increasing a model's size brings difficulties to the training phase (it needs from more training samples increasing training time and demands high computational resources). Note that, under this approach, the number of input neurons depends on the number of monitoring samples in the time interval defined by parameter *w*. In the numerical results in Section 4, we provide a comparative analysis of both *feature-based* and *data-based* approaches.

D. Decision Making

Finally, let us to sketch the main algorithm in the decision-making module (Table II) that makes decisions upon the arrival of predictions for the short and long terms. As presented in Fig. 1, the decision maker receives the estimation of both predictors: *i*) short-term prediction (*st*) with the number of LDPC iterations needed in next δ_s time period; and *ii*) long-term prediction (*lt*) with the prediction of BER threshold violation in the next δ_l time period. Moreover, it receives the current state of the system and the number of LDPC iterations currently configured in the receiver.

Table II. Decision making algorithm

```
INPUT: st(δs), lt(δl), state, FEC_ite
OUTPUT: state
next=findNextState(st(δs), lt(δl))
if next = "critical" then
    if state != "critical" then
        FEC_ite ← maxIte
        sendNotif('thrExc',0)
    return next
else
    if state = "critical" then
        sendNotif('thrDec',δl)
    if FEC_ite<st(δs) then
        reconfigureFEC(st(δs), 0)
    else if FEC_ite>st(δs) then
        reconfigureFEC(st(δs), δs)
return next
```

Table III. ATA States Definition

Next State	$st(\delta_s)$	$lt(\delta_l)$
normal	== *minIte*	==*belowThr*
warning	> *minIte*	==*belowThr*
critical	*	==*aboveThr*

The very first step of the decision-making process is to compute the next state according to the input data (line 1 in Table II). In this regard, we devise the three options summarized in Table III. The system will go into *normal* state if *st* equals *minIte* and no BER threshold violation is predicted. This can be the case in absence of fiber stressing events and any other soft-failure. On the contrary, if *st* exceeds *minIte* but below BER threshold, the system is going to enter a *warning* state. Note that this is exactly the situation depicted in Fig. 2a. Finally, if BER threshold violation is anticipated, no matter the short-term prediction is, the system enters a *critical* state, which corresponds to the example in Fig. 2b.

The rest of the decision-making algorithm compares the current and the next state to take some additional actions. In particular, if the system enters into *critical* state, two consecutive actions are taken: *i*) the number of LDPC iterations is set to *maxIte*, to guarantee even more QoT assurance; and *ii*) a notification is immediately sent to the node controller to notify the predicted BER threshold violation (lines 2-6). In case that conditions allow changing from *critical* to any other state, firstly a notification is scheduled for next δ_l time to inform about this (lines 7-9). Then, the short-term prediction is used to reconfigure the receiver and different strategies are followed depending on the current and forecasted number of iterations. If more iterations are required, the reconfiguration is done immediately; on the contrary, the reconfiguration is delayed δ_s time units (lines 10-13). In this way, robust and conservative reconfiguration is achieved. Finally, the next state is returned, which becomes the current state once new predictions are available.

4 RESULTS

In this section, we first describe the experimental testbed and then give details about how data for training and testing ML models have been generated. Next, the results from exhaustive numerical analysis on the ML-based predictors used in different ATA modules is presented highlighting those configurations that achieve maximum transmission robustness. Finally, a numerical analysis on expected power consumption savings for an illustrative experimental case is presented.

A. Experimental testbed and data generation

In this section, we detail the experimental assessment used to generate real data to train and validate short-term and long-term predictors.

The SOP and the corresponding dataset have been collected from an experimental setup that includes a real-time differential quadrature phase shift keying (DQPSK) optical coherent receiver operating at 7 Gbaud after 50km of single- mode fiber (SMF) and optical noise loading. Mechanical events are caused by a programmable robot arm twisting the fiber and the initial SOP is randomly rotate before each acquisition. The receiver, shown in Fig. 5, consists of low resolution 5-bit ADCs with 3.4 effective number of bits at 7 GHz and a sampling rate of 14 GSa/s after an integrated coherent mixer and followed by field-programmable gate arrays (FPGA); digital signal processing (DSP) is carried out in one main FPGA.

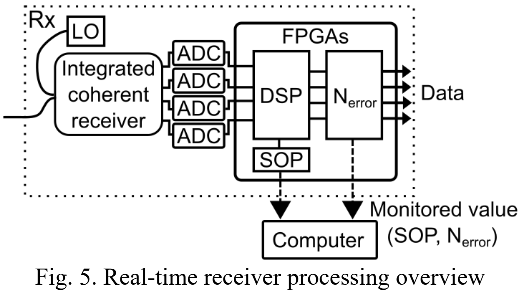
Rx
LO
Integrated coherent receiver
ADC
ADC
ADC
ADC
FPGAs
DSP
N$_{error}$
Data
SOP
Computer
Monitored value (SOP, N$_{error}$)

Fig. 5. Real-time receiver processing overview

Every experiment is triggered by rotation speed threshold and contains a number of monitoring samples before and after the triggering mechanical event. SOP monitoring samples, extracted from the de-multiplexed polarization steps in the DSP are gathered at a rate of 3,600 samples per second, i.e., 1 sample every 278 µs. (see details in [11]). In addition, the number of errors is evaluated by counting the difference between the expected pseudorandom binary sequence of length 2^{11} and the decoded one (by steps of 32 bits) on each tributary. The number of error extraction rate is synchronized with the SOP extraction rate. Pre-FEC BER is then computed by considering the baud rate, the polarization multiplexed factor and the number of bits.

Due to the limited space in the FPGA, the FEC was not implemented and further consideration will be based on a *toy model* linking optical signal on noise ratio, pre-FEC BER and final block errors; it is based on a concatenation of a LDPC followed by a Bose–Chaudhuri–Hocquenghem code to avoid error floors [25]. As implementation penalty of the receiver does not allow us to work at a real BER regime, we build a model with shifted differential gain compared to a theoretical approach (e.g., [26]) to circumvent this limitation.

The experimental testbed was used to generate data for ML training and testing. To that end, we run more than 500 different experiments triggered by mechanical events of different magnitudes (twisting between 20 and 40 degrees per second); every experiment generated 2,000 monitoring samples before the triggering event and 14,000 samples after it, lasting around 5 seconds per experiment. From all the experiments, 150 of them were selected for ML model training and testing purposes providing a wide range of SOP and pre-FEC BER fluctuations. Selection of samples was manually done to eliminate redundant samples, as well as to keep moderate the size of the datasets to manage. Next, raw experimental data was aggregated by averaging samples at intervals of 10 ms. Then, modelling samples were created by either computing *cur* and *trend* features in equations (1)-(2) or storing the whole aggregated data (for *data-based* ANN) in the time window [t-w+1, t]. Finally, the actual value for the response variable at $t+\delta$ was computed and assigned to each modelling sample.

Aiming at finding the relation between the number of LDPC iterations and the maximum pre-FEC BER that can be corrected with no post-FEC errors, an additional experiment was carried out to calibrate the system with our toy model. Fig. 6 shows the obtained results (dashed line) and the considered pre-FEC BER ranges used to label samples (solid segments). The reason why segments do not overlap with the experimental curve is because we considered a certain *surplus* (10% in the case of Fig. 6) to skip labelling samples under the actual number of iterations required.

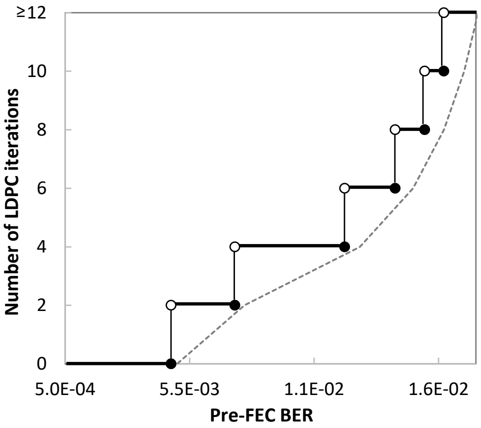

Fig. 6. Number of iterations vs pre-FEC BER

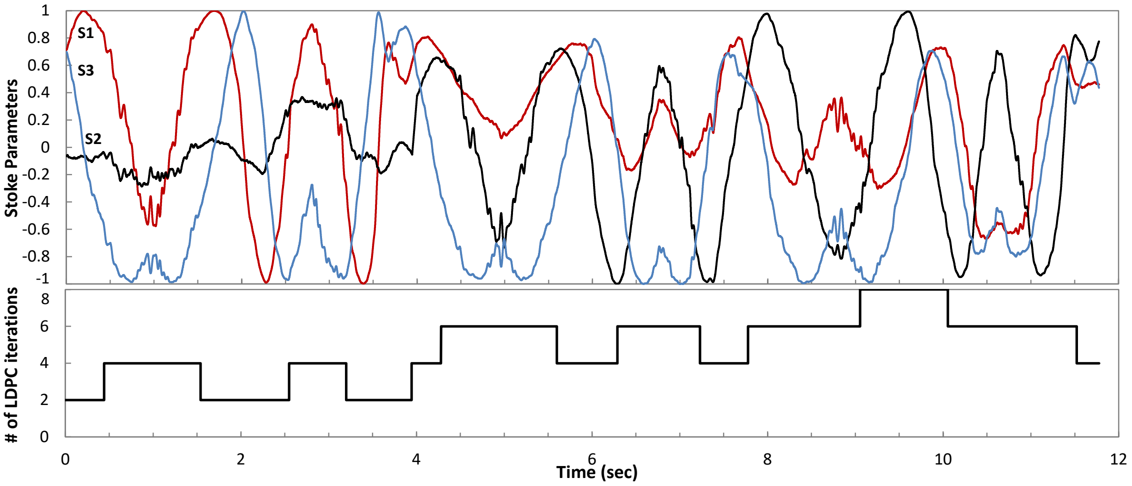

Fig. 7. SOP fluctuations and required number of LDPC iterations vs. time.

For illustrative purposes Fig. 7 shows an example of a concatenation of experiments with strong SOP fluctuations leading to variations in the response variable for the specific case of the short-term predictor, where δ=10 ms. We observe that the amplitude and speed of SOP fluctuations is very high, as oscillations in the full range [-1, 1] can be observed in just few seconds. Variations on the Poincare Sphere are represented in Fig. 8. In addition, SOP variations are not simply related to the number of LDPC iterations, thus clearly motivating the need of a ML model to find accurate correlation between input data and response.

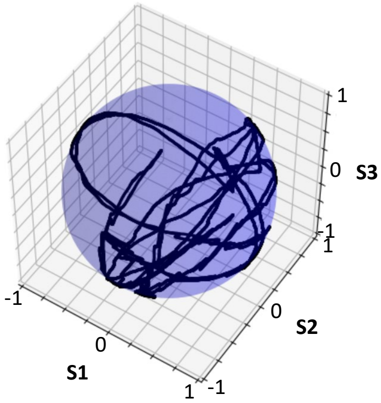

Fig. 8. SOP fluctuations represented on the Poincare sphere

B. *Predictors numerical analysis*

Once data generation has been detailed, let us now focus on evaluating the accuracy of the distinct ML models for the short-term predictor. Among all different ways to compute accuracy, we selected the *sub-estimation error* as the most critical one, as it quantifies the percentage of decisions where less iterations than those actually needed are forecast. Note that these errors could lead to wrong receiver configuration causing connection disruption due to non-zero post-FEC BER. In order to amplify the effect of this error, we labelled the response class assuming 0% surplus. As for the utilization of data, we randomly split them in 80% for ML model training and 20% for testing, considering several runs to perform cross-validation.

Fig. 9a shows the accuracy of all the proposed ML-models. Aiming at numerically comparing the new contributions with our preliminary work in [20], we start by considering only SOP as input data. Thus, the NB model matches with the unique ML modelling approach presented in [20]. Accuracy results are presented for training and testing data independently, and *w*=200 ms. The very first conclusion that can be drawn is that ANN modelling exhibits higher robustness than NB. Although NB shows the best accuracy in training data, the poor performance in predicting testing samples makes unappropriated its practical application in real scenarios where not all relevant events could be reproduced during the initial training phase. Note that that the undesired performance could be mitigated by adopting sophisticated re-training strategies improving models during operation as soon as new events are monitored and learned, as proposed in [27] and [28]. However, the number of samples used for training is large, which denotes the rigidity to extrapolate of this model beyond the observations.

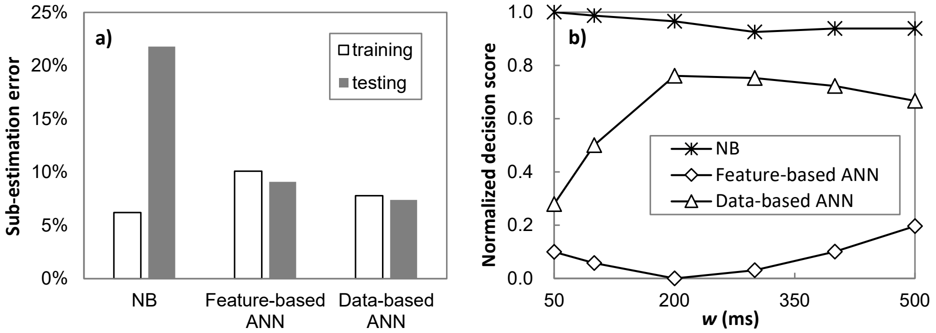

Fig. 9. Short-term predictor analysis: a) accuracy, b) model selection

On the contrary, ANN-based approaches produce more robust performance in both training and testing sample datasets. In terms of accuracy, *data-based* ANN slightly improves *feature-based* one. However, the model size significantly differs between both approaches and therefore, a critical question to answer is whether the improvement in accuracy of the *data-based* model justifies its larger size compared to that of the *feature-based* one. In order to solve this question, we defined a decision score as the number of sub-estimation errors plus the logarithm of the number of model coefficients (the smaller the better), following a similar approach than other model selection indicators [29]. Fig. 9b shows the value of that indicator (conveniently normalized) for all modelling approaches and different configurations of parameter *w*. We observe that *feature-based* ANN provide the best trade-off between accuracy and model size; it reaches the minimum value for *w*=200 ms. In light of this, we conclude that *feature-based* ANN is the most proper approach for ML-based modelling and hence, this approach is selected hereafter.

Once a ML model has been selected, our target is to achieve a model with no sub-estimation errors, which guarantees target QoT. To this end, we studied the impact of adding pre-FEC BER information as input data to the model. Moreover, configuring a proper *surplus* value during training helps to achieve the objective. Specifically, by increasing the value of the surplus, sub-estimation error decreases at the expense of exceeding the actual needed number of iterations and consequently, reducing power consumption savings. The impact of those effects can be observed in Fig. 10; *feature-based* ANN was trained considering either SOP data or pre-FEC BER data or both, for a range of surplus values. We note that using SOP and Pre-FEC BER data remarkably improves model performance compared to only using one of them. Indeed, a *surplus* value as low as 5% allows producing predictions with zero sub-estimation error.

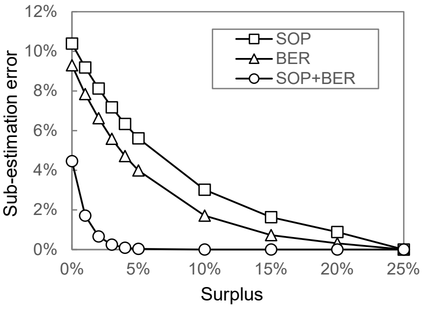

Fig. 10. Sub-estimation error reduction

Let us eventually to evaluate the accuracy for a long-term predictor aiming at predicting whether BER threshold is violated. To this aim, training samples were conveniently re-labeled; particularly, those providing an expected pre-FEC BER higher than 1.5 10^{-2} were labeled *aboveThr*. Using the above-described *feature-based* ANN configuration, a long-term predictor was trained considering a wide range of δ values. Fig. 11 plots the actual threshold anticipation time (i.e., when the model correctly identifies an *aboveThr* event in the horizon) as a function of δ targeting at showing how long δ can be to guarantee an accurate threshold violation detection.

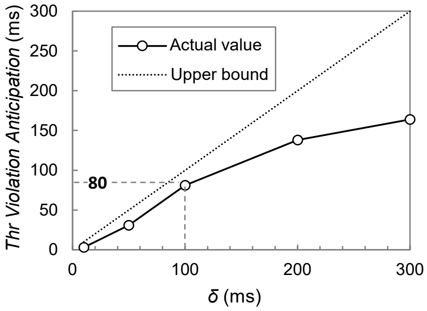

Fig. 11. Long-term predictor analysis

Although models could potentially target actual anticipation time equal to δ (dashed line), this becomes slightly smaller since increasing δ makes it more difficult to detect threshold violation, which delays the anticipated detection. The highlighted point show that when expected anticipation of 100 ms is targeted, effective 80 ms actual threshold anticipation is achieved. For larger δ values, detection time drops and therefore, maximum δ=100 ms is proposed for accurate long-term predictors to anticipate threshold violation. Note that 80ms leaves enough time to trigger some reconfiguration procedure at the location of the transmitter.

C. Power consumption savings

The last set of numerical results is oriented to quantify the power consumption reduction that our proposed ATA in the local receiver subsystem can achieve compared to a static configuration guaranteeing transmission robustness. Table IV summarizes the average number of iterations and power consumption of the LDPC module according to the model in [6], [30] for the three ATA states defined for decision making in Section 3.D and observed in the experimental data. We can see that only under the *critical* state (i.e., close to the pre-FEC BER threshold), ATA power consumption is that of the static configuration. For any other scenario, even in the *warning* state (i.e., fluctuations causing number LDPC iterations to increase), power consumption is noticeably reduced due to the small average number of LDPC iterations actually required.

Table IV. ATA performance

	Static config	**ATA** *normal*	*warning*	*critical*
# LDPC iterations	10	2	2.18	10
Power consumption (W)	25	3.5	6.92	25

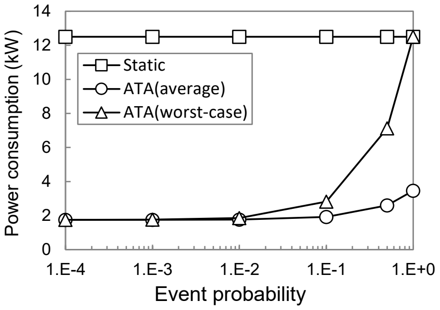

Fig. 12. Power consumption savings

According to the results in Table IV, we can easily do numerical results, like the one represented in Fig. 12, where the operation of a network supporting 250 optical connections (500 transceivers) was simulated; each transceiver is equipped with ATA in the receiver. We assume aerial fibers subject to fiber stressing events caused by environmental disturbances, such as strong wind [31]. Fig. 12 shows total power consumption as a function of the probability (p) of such fiber stressing events requiring LDPC iterations to increase. In particular, three cases are compared: *i*) the static configuration where the conservative maximum number of iterations is setup; *ii*) the average operation of ATA under those events observed in the *warning* state; and *iii*) a worst case where all events lead to the *critical* state. Note that the latter meets the static one in the (unlikely) case of p=1. Let us now put the spotlight on a remarkably large but realistic probability p=0.01, which could correspond with a windy day where few gusts of high wind per day are usually measured [32]. In that case, ATA allows achieving power consumption savings around 80% compared to the static operation. Indeed, those large power consumption savings are expected for large periods of time since fiber stressing event probability is frequently smaller than 0.01. Note that, according to the functional power distribution analysis in [33], overall DSP power consumption savings can reach 20% for metro network scenarios. Since ATA also guarantees zero post-FEC errors, we can finally validate our proposed ATA as a smart solution towards efficient and adaptive autonomic transmission.

5 CONCLUDING REMARKS

The concept of autonomic transmission has been realized by means of an autonomic transmission agent (ATA) running inside of coherent transponders. The agent is based on ML and focuses on anticipating pre-FEC BER degradation of individual lightpaths by monitoring real time the Stokes parameters representing the SOP in addition to the current pre-FEC BER. The loop can be closed locally by tuning the number of LDPC iterations of the soft-decision FEC process with the objective of minimizing power consumption at the receiver by operating at the minimum number of LDPC iterations that guarantees zero post-FEC error transmission. Alternatively, the loop can be closed remotely through the SDN controller in case the predicted pre-FEC BER is over a given threshold.

Two different approaches to build predictive ML-based classifiers for short-term and long-term prediction were explored. On the one hand, a Naïve-Bayes approach was used to map a discretized version of the input features (i.e., the current value and trend of SOP and pre-FEC BER) with the target class of the response variable (i.e., number of iterations for the short-term and expected QoT evolution for the long-term predictor, respectively). In addition, ANNs were trained to predict the response either as a function of the features used in the Naïve-Bayes approach or as a function of the raw monitoring data collected without applying prior feature computation.

Experimental data was obtained from a testbed and used to evaluate the proposed ATA. The testbed included a SMF fiber connected to a DQPSK coherent receiver was stressed mechanically by events caused by a programmable robot arm. After collecting a large number of data samples and training the different ML alternatives, we evaluated the different approaches and concluded that using small ANN with input features summarizing the current value and evolution of both SOP and pre-FEC BER leads to the best results. Note that running time needed to make a prediction with small feature-based ANN is shorter and this, hence, enables ATA for practical in-field operation.

Finally, the impact in terms of power consumption reduction of the proposed ATA operation has been analyzed in comparison to a baseline operation where LDPC iterations are statically configured to guarantee zero post-FEC BER in the event of any observed fiber stressing event. The comparative results showed power consumption savings as large as 80% and 20% for FEC and overall DSP, respectively, even for the very unlikely scenario where fiber stressing events probability was large (>10%).

In view of the obtained results, we can conclude that the proposed ATA allows for robust operation of low-margin networks with low-cost transmission hardware and low-power consumption operation. As a final remark, note that the ATA requires from an exhaustive training phase to ensure zero post-FEC errors. Such training can be initially carried out out-of-field with a large dataset containing samples from the field that can be complemented with data from simulation and lab experiments. Then, once deployed in the field, re-training can be carried out to improve the models as soon as new events are monitored and learned, as proposed in [27].

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Commission for the H2020-ICT-2016-2 METRO-HAUL project (G.A. 761727), from the AEI/FEDER TWINS project (TEC2017-90097-R), and from the Catalan Institution for Research and Advanced Studies (ICREA).

REFERENCES

[1] Cisco Visual Networking Index: Forecast and Trends, 2017–2022.

[2] Y. Pointurier, “Design of Low-Margin Optical Networks,” IEEE/OSA Journal of Optical Communications and Networking, vol. 9, pp. A9-A17, 2017.

[3] L. Velasco, P. Wright, A. Lord, and G. Junyent, “Saving CAPEX by Extending Flexgrid-based Core Optical Networks towards the Edges,” (Invited Paper) IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 5, pp. A171-A183, 2013.

[4] T. Kupfer, A. Bisplinghof, T. Duthel, C. Fludger, S. Langenbach, “Optimizing Power Consumption of a Coherent DSP for Metro and Data Center Interconnects,” in Proc. OFC 2017.

[5] X. Chen, S. Chandrasekhar, S. Randel, W. Gu, P. Winzer, “Experimental Quantification of Implementation Penalties from Limited ADC Resolution for Nyquist Shaped Higher-Order QAM,” in Proc. ECOC, 2016.

[6] C. Dorize, O. Rival, and C. Costantini, “Power scaling of LDPC decoder stage in long haul networks,” in Proc. PS, 2012.

[7] J. Park and K. Chung, “An adaptive low-power LDPC decoder using SNR estimation,” EURASIP Journal on Wireless Communications and Networking, vol. 48, pp. 2-9, 2011.

[8] J. Auge, “Can we use Flexible Transponders to Reduce Margins?” in Proc. OFC 2013.

[9] I. Sartzetakis, K. Christodoulopoulos, and E. Varvarigos, “QoT Aware Adaptive Elastic Optical Networks,” in Proc. OFC, 2017.

[10] A. P. Vela, B. Shariati, M. Ruiz, F. Cugini, A. Castro, H. Lu, R. Proietti, J. Comellas, P. Castoldi, S. J. B. Yoo, and L. Velasco, “Soft Failure Localization during Commissioning Testing and Lightpath Operation [Invited],” IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 10, pp. A27-A36, 2018.

[11] F. Boitier, V. Lemaire, J. Pesic, L. Chavarria, P. Layec, S. Bigo, E. Dutisseuil, “Proactive fiber damage detection in real-time coherent receiver,” in Proc. ECOC, 2017.

[12] D. Rafique and L. Velasco, “Machine Learning for Optical Network Automation: Overview, Architecture and Applications,” IEEE/OSA Journal of Optical Communications and Networking, vol. 10, pp. D126-D143, 2018.

[13] A. P. Vela, M. Ruiz, L. Velasco, “Distributing Data Analytics for Efficient Multiple Traffic Anomalies Detection,” Elsevier Computer Communications, vol. 107, pp. 1-12, 2017.

[14] R. Casellas, R. Martínez, R. Vilalta, R. Muñoz, “Control, Management and Orchestration of Optical Networks: Evolution, Trends and Challenges,” IEEE/OSA Journal of Lightwave Technology, vol. 36, pp. 1-13, 2018.

[15] L. Velasco, A. Chiadò Piat, O. González, A. Lord, A. Napoli, P. Layec, D. Rafique, A. D'Errico, D. King, M. Ruiz, F. Cugini, and R. Casellas, “Monitoring and Data Analytics for Optical Networking: Benefits, Architectures, and Use Cases,” IEEE Network Magazine, vol. 33, pp. 100-108, 2019.

[16] L. Velasco, A. P. Vela, F. Morales, and M. Ruiz, “Designing, Operating and Re-Optimizing Elastic Optical Networks,” (Invited Tutorial) IEEE/OSA Journal of Lightwave Technology (JLT), vol. 35, pp. 513-526, 2017.

[17] Ll. Gifre, J.-L. Izquierdo-Zaragoza, M. Ruiz, and L. Velasco, “Autonomic Disaggregated Multilayer Networking,” IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 10, pp. 482-492, 2018.

[18] L. Velasco, Ll. Gifre, J.-L. Izquierdo-Zaragoza, F. Paolucci, A. P. Vela, A. Sgambelluri, M. Ruiz, and F. Cugini, “An Architecture to Support Autonomic Slice Networking [Invited],” IEEE/OSA Journal of Lightwave Technology, vol. 36, pp. 135-141, 2018.

[19] L. Velasco, A. Sgambelluri, R. Casellas, Ll. Gifre, J.-L. Izquierdo-Zaragoza, F. Fresi, F. Paolucci, R. Martínez, and E. Riccardi, “Building Autonomic Optical Whitebox-based Networks,” IEEE/OSA Journal of Lightwave Technology, vol. 36, pp. 3097-3104, 2018.

[20] B. Shariati, F. Boitier, M. Ruiz, P. Layec, and L. Velasco, “Autonomic Transmission Through pre-FEC BER Degradation Prediction Based on SOP Monitoring,” in Proc. European Conference on Optical Communication (ECOC), 2018.

[21] B. Shariati, M. Ruiz, J. Comellas, and L. Velasco, “Learning from the Optical Spectrum: Failure Detection and Identification [Invited],” IEEE/OSA Journal of Lightwave Technology, vol. 37, pp. 433-440, 2019.

[22] A. P. Vela, M. Ruiz, F. Fresi, N. Sambo, F. Cugini, G. Meloni, L. Potí, L. Velasco, and P. Castoldi, “BER Degradation Detection and Failure Identification in Elastic Optical Networks,” IEEE/OSA Journal of Lightwave Technology, vol. 35, pp. 4595-4604, 2017.

[23] P. Sanjiban, S. Roy, and V. Balas, Handbook of Neural Computation, 1st Edition, Elsevier, 2017.

[24] M. Li, T. Zhang, Y. Chen, and A. Smola, “Efficient mini-batch training for stochastic optimization,” in Proc. of ACM KDD, 2014.

[25] K. Sugihara, Y. Miyata, T. Sugihara, K. Kubo, H. Yoshida, W. Matsumoto, and T. Mizuochi, “A spatially-coupled type LDPC code with an NCG of 12 dB for optical transmission beyond 100 Gb/s,” in Proc. OFC, 2013.

[26] A. Rasmussen, M. Yankov, M. Berger, K. Larsen, and S. Ruepp, “Improved Energy Efficiency for Optical Transport Networks by Elastic Forward Error Correction,” IEEE/OSA Journal of Optical Communications and Networking, vol. 6, pp. 397-407, 2014.

[27] L. Velasco, B. Shariati, F. Boitier, P. Layec, and M. Ruiz, “A Learning Life-Cycle to Speed-up Autonomic Optical Transmission and Networking Adoption,” IEEE/OSA Journal of Optical Communications and Networking, vol. 11, 2019.

[28] M. Ruiz, F. Tabatabaeimehr, and L. Velasco, “Knowledge Management in Optical Networks: Architecture, Methods and Use Cases [Invited],” IEEE/OSA Journal of Optical Communications and Networking, vol. 12, pp. A70-A81, 2020.

[29] H. Bozdogan, “Model selection and Akaike's Information Criterion (AIC): The general theory and its analytical extensions,” Psychometrika, vol. 52, pp. 345-370, 1987.

[30] M. Dongsheng and R. Bondade, “Enabling Power-Efficient DVFS Operations on Silicon”, IEEE Circuits and Systems Magazine, vol. 10, pp. 14-30, 2010.

[31] J. Wuttke, P. M. Krummrich, and J. Rosch, “Polarization oscillations in aerial fiber caused by wind and power-line current,” IEEE Photonics Technology Letters, vol. 15, pp. 882-884, 2003.

[32] AEMET OpenData, https://opendata.aemet.es/

[33] T. Kupfer, A. Bisplinghof, T. Duthel, C. Fludger, and S. Langenbach, “Optimizing Power Consumption of a Coherent DSP for Metro and Data Center Interconnects,” in Proc. OFC, 2017.