Deep Learning -based Real-Time Analysis of Lightpath Optical Constellations [Invited]

M. RUIZ, D. SEQUEIRA, AND L. VELASCO*

Universitat Politècnica de Catalunya, Barcelona, Spain
**Corresponding author: luis.velasco@upc.edu*

Received XX Month XXXX; revised XX Month, XXXX; accepted XX Month XXXX; posted XX Month XXXX (Doc. ID XXXXX); published XX Month XXXX

Optical network automation requires accurate physical layer models, not only for provisioning but also for real-time analysis. In particular, In-Phase (I) and Quadrature (Q) constellation analysis enables deep understanding of the characteristics of optical connections (lightpaths), e.g., their length. In this paper, we present methods for modeling lightpaths based on deep learning. Specifically, we propose using autoencoders (AE) and deep neural networks (DNN). Models are trained and composed in a sandbox domain with the information received from the network controller and sent to the node agent that uses them to compare the features extracted from the received signal and the expected features returned by the models. We investigate two different use cases for lightpath analysis focused on lightpath length and optical signal power. The results show a remarkable accuracy for the lightpath modelling and length prediction and a noticeable performance of the AEs for unsupervised IQ constellation features extraction and relevance analysis.

http://dx.doi.org/10.1364/JOCN.99.099999

1. INTRODUCTION

The new type of services enabled by 5G and beyond technologies require to be supported by a programmable autonomous optical network infrastructure [1, 2]. Such new types of services usually require high network performance in terms of bitrate and latency [3], which, together with their dynamicity, makes decentralization and softwarization essential features of that infrastructure.

Monitoring and real-time data analytics are key enablers for the realization of network automation [4]. In particular, Artificial Intelligence (AI) and especially, Machine Learning (ML) -based algorithms have been extensively applied to optical communications to enhance their overall performance [5]. Applications include identifying and predicting optical transmission parameters to mitigate different physical layer impairments, including both linear interference (LI), e.g., Amplified spontaneous emission (ASE) noise, and nonlinear interference (NLI) noise caused by the Kerr effect. In fact, one of the most active fields of application in optical networks is for optical performance monitoring [6] and particularly interesting are those ML-based models that combine the characteristics of the physical system and real-time monitoring data to produce accurate estimation of NLI noise [7, 8]. Other approaches receiving large attention are those exploring deep learning (DL) techniques to extract information from complex, dense monitoring data inputs, without knowledge of the physical characteristics. E.g., in optical coherent systems with advanced Digital Signal Processing (DSP) techniques, the analysis of In-Phase and Quadrature (IQ) optical constellation diagrams as images can be performed by means of training convolutional neural networks to estimate the Quality of Transmission (QoT) of optical signals [9].

AI/ML/DL techniques usually require large data sets for training purposes. Although such data should come from experimental setups, much research is being carried out using accurate simulation environments based on analytical models (e.g., VPIphotonics [10] and GNPy [11]). Further, some ML applications use expected signals as reference, e.g., to detect degradations [12]. On the other hand, not only accurate ML models need to be trained, but also precise knowledge of input physical parameters is needed. Examples include the length of the optical connection (*lightpath*) from Transmitter (Tx) to Receiver (Rx) and the Tx launch power. Although some physical parameters might vary with time [13], considering them as inputs of ML models strongly increases the applicability of those models to real scenarios [14]. In this regard, a possible approach is to model individual network elements and concatenate them to create one model for the complete lightpath. Note that model concatenation is a common approach, which is part of other ML techniques, e.g.,

autoencoders (AE) [15]; AEs are a type of deep neural networks (DNN) with two network components: the *encoder* and the *decoder*.

In our previous work in [16], we proposed a method for IQ constellation analysis based on AEs. Our AE models run at the Rx site and generate compressed constellation data, lightpath metrics estimation, and additional metrics obtained after applying *explainable AI* techniques. Although AEs themselves require training, the encoder transforms the input in a reduced number of *latent features* in an unsupervised manner, so that the decoder is able to reconstruct the original input from the latent features space.

In this paper, we propose a comprehensive solution for in-operation lightpath analysis of IQ constellations. Specifically, the contribution of this paper is two-fold:

- A novel network functional architecture is presented in Section 2. A sandbox domain is used to obtain DL models suited for the lightpath under analysis. Models include AEs, statistical distributions of IQ constellation points, and DNN-based lightpath metric predictors. The models are designed to run at the Rx site, and continuously analyze lightpath's metrics to compress monitored constellation samples and detect potential anomalies.
- A methodology for constellation analysis based on Gaussian Mixture Models (GMM) [17] (supervised) and AE-based (unsupervised) feature extraction is presented in Section 3. Moreover, a lightpath modelling approach consisting in concatenating DNN models emulating the performance of optical components, e.g., Reconfigurable Optical Add-Drop Multiplexers (ROADMs) and optical links including intermediate Optical Amplifiers (OA), is presented and used to generate expected constellations and synthetic constellation samples.

Two different use cases of lightpath analysis using the proposed constellation analysis and lightpath modelling methodology are detailed in Section 4. Specifically, path length and power analysis using GMM-based and AE-based constellation analysis are proposed. The numerical evaluation of the different methods and algorithms is presented in Section 5. A simulation environment reproducing a realistic network deployment is used to generate synthetic IQ constellation samples for a wide range of lightpath configurations. Finally, Section 6 concludes the paper.

2. AI-BASED CONSTELLATION ANALYSIS

Fig. 1 overviews the considered network architecture and will be used for describing the main workflow; for the sake of simplicity, only the directly involved elements, like a lightpath, a node controller and a sandbox are detailed, whereas other components have been sketched -e.g., the Software Defined Networking (SDN)- or omitted to better highlight the key concepts involved in this work. Lightpath *i* is considered as the entity under analysis, which is represented as a sequence of optical components (Tx, ROADMs, links, and Rx) supporting that lightpath. Fig. 2 details the internal architecture of the sandbox domain and node agents.

At set-up time, the SDN controller solves the Routing, Spectrum, and Transponder Assignment [19] before configuring the involved devices to establish the lightpath. Then, after the lightpath is provisioned, the sandbox domain receives the lightpath's configuration from the SDN controller (labeled 1 in Fig. 1), including its route on the optical network and some metrics. This configuration is used in the sandbox domain to set up an accurate representation of that lightpath to be set in the Rx agent (2 in Fig. 1). Such representation is defined as a sequence of pre-trained DL-based models that emulate the behavior of each individual optical component that the optical signal traverses. The role of the models is different depending on the physical element they characterize (Fig. 2a). For instance, the model for the Tx characterizes the output signal according to its specifications, whereas the models for intermediate elements (ROADMs and fiber links) propagate forward a set of features related to the signal's constellation. Specifically, intermediate components introduce distortion on the constellation as a result of LI and NLI noise. Finally, the Rx model receives the constellation features and performs additional actions before returning the

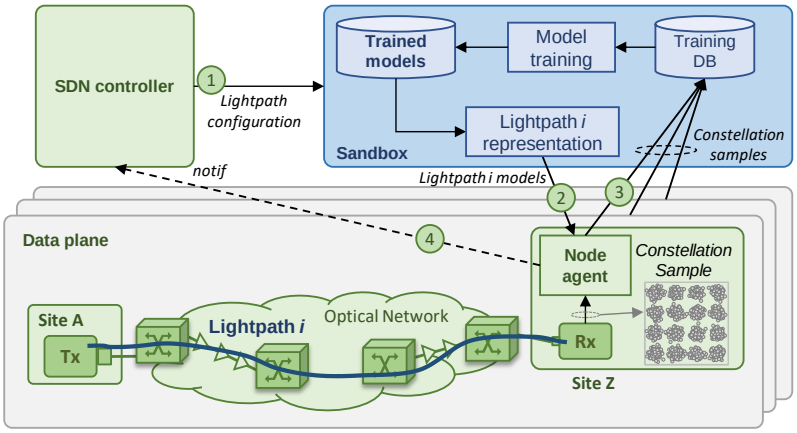

Fig. 1. Reference network architecture.

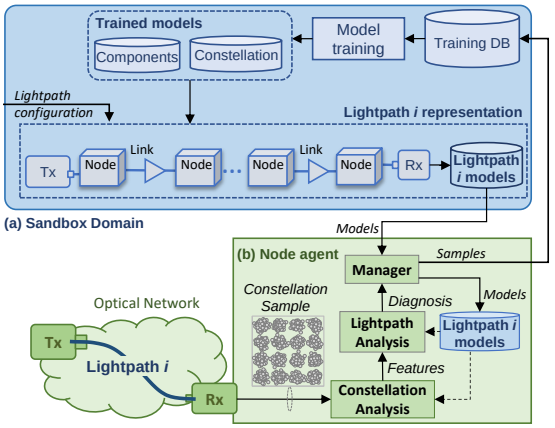

Fig. 2. Details of sandbox (a) and node agent (b).

output of the model. Additionally, any relevant change affecting the lightpath during its life-time, e.g., path rerouting, needs to be notified to the sandbox, so as to adapt the lightpath's representation and avoid misleading diagnosis due to mismatch between the physical lightpath and its models.

Both constellation and lightpath analysis require from models that characterize the monitored lightpath. Thus, anomaly detection based on comparing observed features and expected ones coming from lightpath's models can be carried out at the Rx. Note that this scheme highly reduces the amount of data to be sent to the centralized elements (3 in Fig. 1), as well as its computational demand for real-time data analysis purposes [18]. Once the lightpath is set-up and the lightpath models set in the Rx agent, they are used for analysis. With a predefined frequency, e.g., every 1s, the Rx samples the received constellation and gathers *n* IQ symbols. The sample is then processed by the *constellation analysis* block in the Rx agent [4] (Fig. 2b). The aim of this block is to extract a set of relevant constellation *features* that facilitates posterior analysis, as well as compressing constellation data to be used for multiple purposes, such as model training. These features are obtained by means of both supervised and unsupervised statistics and ML-based techniques (details are given in Section 3). Next, the *lightpath analysis* block processes the features extracted from the received constellation and analyzes key lightpath's configuration metrics, such as length and/or power configuration. The result of this analysis produces a diagnostic report highlighting, e.g., whether some of the metrics does not follow the expected behavior. The diagnostic report is processed by the *manager* block that implements a set of rules and generates notifications to the SDN controller depending on the diagnosis (4 in Fig. 1).

Following the above generic architecture, Fig. 3 illustrates two different use cases for lightpath analysis. The first use case is devoted to checking whether the real length of a given lightpath matches with the expected one (Fig. 3a). This analysis is based on the fact that both LI and NLI noise increase with path length and additionally, both affect the magnitude and shape of the dispersion of the symbols around the expected constellation points. Therefore, differences can be found by comparing the features of the observed constellation points, extracted with supervised techniques, and the expected ones. A model is used to detect any significant difference, as well as to estimate the real length.

As an example, the received constellation of a 16QAM signal is represented in Fig. 3a, where a constellation point (3+3i) is zoomed in. External constellation points get more affected by the NLI noise since not only their shape becomes more dispersed around the central point as it happens in presence of LI noise, but they also become more elliptical, with eccentricity and direction of the axes that depend on the traveled distance [8]. Let us assume that the expected features characterize a distribution that is larger and more elliptical than the one observed. This will be detected by the length analysis module as an anomaly in path length, specifically as

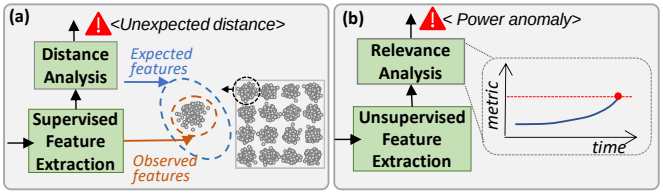

Fig. 3. Lightpath analysis use cases.

a length shorter than expected. The reasons behind that anomaly can be multiple, e.g., inaccurate lightpath configuration, wrong sandbox domain model configuration, lack of synchronization between SDN controller and sandbox after a path rerouting, just to mention a few. Upon the notification of the length analysis module, the SDN controller can trigger the needed procedure to detect and isolate the actual reason for the detected anomaly.

The second example explores a different approach to detect anomalies affecting the launch power at the Tx side (Fig. 3b). Instead of characterizing the gathered constellation through features with the distribution of the constellation points, this use case leverages AEs to compress constellation samples into a reduced set of latent space features in an unsupervised way. The analysis is performed in both forward and backward directions through the AEs to quantify relevance metrics at both the latent feature space and the input. This relevance can be tracked over time so as to detect any drift or shift directly related to a power anomaly (e.g., power drop) in the Tx. Note that an early detection of power anomaly can prevent degradations [13].

3. CONSTELLATION ANALYSIS AND MODELLING

In this section, we present the main procedures for constellation analysis and lightpath modelling. In the following, we are going to consistently denote $X=\{x_1,\ldots, x_n\}$ as an optical constellation sample consisting of *n* IQ symbols. Although each symbol $x_i \in X$ is typically represented as a complex number, for the sake of simplicity, it will be alternatively denoted as a tuple of real values $<x_i^I, x_i^Q>$ with the in-phase x_i^I and quadrature x_i^Q components, respectively. Further, a constellation is defined by a set of constellation points *P*, each identified by its expected centroid $<p_j^I, p_j^Q>$.

A. Supervised and Unsupervised Feature Extraction

Fig. 4 illustrates the feature extraction procedure for a given 16QAM optical constellation sample *X*. The procedure is used to summarize the optical constellation into a number of supervised and unsupervised features; it also produces goodness-of-fit (*GoF*) metrics that allow further evaluation of the quality and usefulness of the generated features.

Let us first focus on the supervised feature extraction approach (Fig. 4a). The objective of this approach is to generate the set of features (also referred as processed sample) *Y* that summarizes *X* with a number of clear, unequivocal, and predefined characteristics. To this aim, we model the constellation points as bivariate Gaussian distributions. This

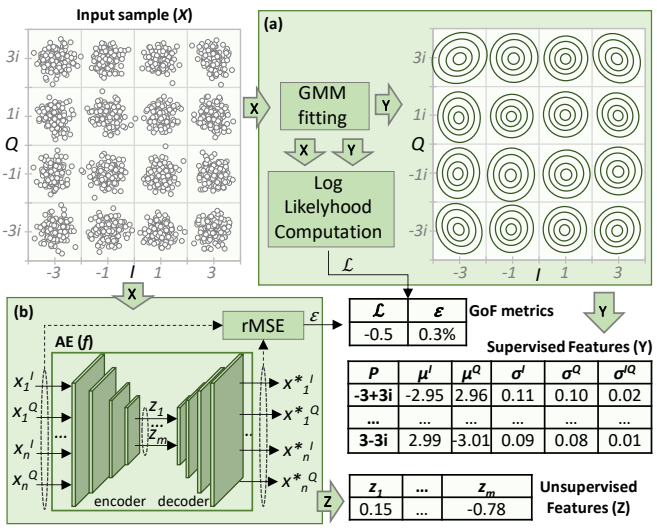

Fig. 4. Feature extraction. Supervised (a) and unsupervised (b).

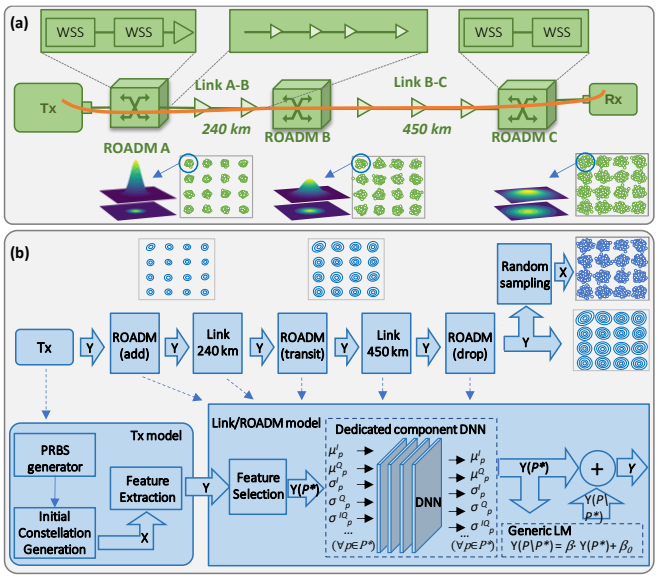

Fig. 5. Lightpath example (a) and its proposed model (b).

approach characterizes every constellation point $p \in P$ with a two-component vector $<\mu^I_p, \mu^Q_p>$ representing the mean position in the constellation and with a three-component vector $<\sigma^I_p, \sigma^Q_p, \sigma^{IQ}_p>$, which captures the variance and symmetric covariance terms that the symbols belonging to the constellation point p experienced around the expected mean.

Aiming at allowing an accurate fitting of each of the constellation points, especially when LI and NLI noise are large and symbols are dispersed far from the expected centroid, we apply GMM fitting for multiple and joint bi-variate Gaussian distribution estimation. GMM is initialized to find $|P|$ different bi-variate Gaussian distributions whose expected centroids are the ones in P. An illustrative example is depicted in Fig. 4a, where the inset values in the table are the features Y computed from the sample and the level curves depict the bi-variate Gaussian distributions.

It is worth noting that by forcing the constellation points to be modelled as Gaussian distributions and by selecting the expected centroids in P as initial points for GMM fitting, the obtained features are strongly conditioned. In order to estimate how accurate the GMM fitting is, i.e., how well they characterize the symbols dispersion around the expected mean, the lowest (worst) likelihood value $\mathcal{L}$ (in logarithmic scale) [20] for one constellation point is returned as GoF metric. The $\mathcal{L}$ metric might have different potential applications depending on the specific use case.

In contrast, unsupervised feature extraction (Fig. 4b) aims at transforming input sample X into a latent sample Z that accurately represents the main characteristics of X without actually defining how to achieve such characterization (e.g., without imposing any statistical distribution). In this case, we use an AE with $2 \cdot n$ inputs for the I and Q components of every symbol in X, followed by a number of hidden layers, each with a number of hidden neurons. The last layer of the encoder component contains m outputs and is commonly known as latent feature space Z. The latent space is the input of the decoder component, which also contains a structure of hidden layers (not necessarily equal to those of the encoder) that lead to a final output layer with $2 \cdot n$ values, each corresponding to one of the initial encoder inputs. The AE is trained using the mean absolute error as loss function (typical for regression applications) so that the error between a given encoder input neuron and its related decoder output is minimized. In this way, the encoder component codes input sample X into latent sample Z, whereas the decoder reconstructs sample Z into the original feature space. Note that the reconstructed sample X^* differs from the original one X; such a difference, that can be quantified in terms of relative mean square error (rMSE), is used as GoF metric (denoted ε) for unsupervised feature extraction.

B. DNN-based Concatenation Modelling

Fig. 5 illustrates the proposed approach to build DL-based lightpath models as a concatenation of DNN-based component models. As illustrated in the example in Fig. 5a, the signal crosses three ROADMs (A, B, C) and optical links with different length and number of spans. ROADMs are modelled with two Wavelength Selective Switches (WSS), and every intermediate ROADM, except the last one before the Rx (drop), includes a booster OA that compensates for WSSs insertion losses. Typically, the insertion losses in the last ROADM are compensated by DSP techniques at the digital coherent Rx. The optical links consist of fiber spans and inline OAs that compensate for the losses of the fiber spans. We assume that the pre-OA at ROADM's input is a part of the link model (see the insets in Fig. 5a).

The concatenation model abstracting the lightpath in Fig. 5a is presented in Fig. 5b. In this case, the ROADMs and optical links in Fig. 5a are modeled using DNNs. The lightpath is modeled as an ordered sequence of: *i*) a Tx model, *ii*) an add ROADM model representing ROADM A, *iii*) a 240-km link model representing fiber link A-B, *iv*) a transit ROADM model representing ROADM B, *v*) a 450-km link

model representing fiber link B-C, and finally *vi*) a drop ROADM model representing ROADM C.

The Tx model includes a pseudo-random bit sequence (PRBS) generator used to generate the initial optical constellation following a Tx configuration. Such initial constellation can be generated using analytical equations, simulation, ML models, etc. Once the initial optical constellation is generated, a feature extraction block computes the supervised features Y as described in the previous section.

Models for both ROADM and link components follow a similar architecture. Those models propagate feature set Y, modifying the mean and variance of each constellation point according to the LI and NLI noise that the physical element introduces. Aiming at reducing the complexity of the DNN models, a subset of relevant constellation points is selected as representative of the impact of noise during propagation, whereas the rest are generated as a function of the propagated points. Hence, we need to first select the reduced set of constellation points $P^* \subset P$. In particular, all the features in Y that belong to constellation point subset P^*, denoted as $Y(P^*)$, are selected. Then, this reduced set of features is propagated through a DNN model specifically trained for the component that is represented. The structure of the DNN consists in $5 \cdot |P^*|$ input and output features (i.e., μ and σ vectors of the selected constellation points), and a number of hidden layers with variable number of hidden neurons each. Since the final outcome of the model must include the whole set of features, a linear regression model mapping the characteristics of the non-selected constellation points, denoted as $Y(P \backslash P^*)$, as a function of selected ones is used. This model is generic and can be shared between components of different types; it is defined by a matrix of linear coefficients β of size $5 \cdot |P \backslash P^*| \times 5 \cdot |P^*|$ and an intercept vector β_0 of length $5 \cdot |P \backslash P^*|$.

The proposed modelling approach can be also used as a lightweight optical system simulator. By generating random samples following the bi-variate Gaussian distributions defined by Y, synthetic constellation samples can be obtained. In Fig. 5b, such random sampling is performed to generate a synthetic optical constellation at the Rx side; however, random sampling can also be applied after any component model, thus generating intermediate constellations. It is worth noting that the time to generate the resulting optical constellation samples (at the Rx and intermediate points) is noticeably short, since it entails propagating values through a set of DNNs, i.e., only a very a limited number of simple calculations is required [21].

4. LIGHTPATH ANALYSIS USE CASES

In this section, we detail the main algorithms to perform the two use cases of lightpath analysis sketched in Section 2. The proposed algorithms use the feature extraction and lightpath modelling procedures detailed in Section 3. Firstly, the algorithm to set up the lightpath models is presented.

Algorithm 1. Lightpath models setup/update

INPUT: *R, DB, params* **OUTPUT:** < *sca*, *ae*, *pme*, $\mathcal{L}_{thr}$>

```
1:  L ← ∅; sca ← ∅; Yhist← ∅
2:  for r∈R do
3:      l ← get(DB['component'], r.attributes)
4:      L ← append(L, l)
5:  add(DB['lightpaths'][R.id], L)
6:  for i = 1..params.nrep do
7:      for l∈L do
8:          if l.type == "tx" then
9:              b ← l.PBRS(params.n)
10:             X ← l.generateConstellation(b)
11:             Y ← GMMfitting(X)
12:         else Y ←l.propagate(Y)
13:     Yhist ← Yhist ∪ Y
14: for <Yi, Yj>∈ Yhist do
15:     val←computeChi2(Yi,Yj) (eq. (1))
16:     if sca == ∅ or sca.thr<val then
17:         sca.ref ← <Yi, Yj>
18:         sca.thr ← val
19: ae ← get(DB['AE'], R.tx.power)
20: pme ← get(DB['predictor], R.tx.power)
21: Lthr ← get(DB['GoF'], R.tx.power)
22: return < sca, ae, pme, Lthr>
```

A. Lightpath Models Setup/Update

Upon lightpath provisioning or the modification of the attributes of an already in-operation one, e.g., because of re-routing, the models for lightpath analysis need to be loaded in the Rx agent (labeled 2 in Fig. 1). Algorithm 1 details such procedure, which runs in the sandbox domain and is triggered every time a notification is received from the SDN controller with the details of the established/modified lightpath (1 in Fig. 1). The algorithm receives as inputs: *i*) the description of lightpath R, including its id and the sequence of nodes and links that composes the lightpath from Tx to Rx, each with its own configuration attributes; *ii*) the connection to the database of trained models (*DB*); and *iii*) a set of configuration parameters. The output of Algorithm 1 is the set of models and parameters needed for lightpath analysis at the Rx site. These output set includes: *i*) parameter $\mathcal{L}_{thr}$ with the minimum threshold to consider constellations points as accurate Gaussian distributions, *ii*) the statistically-based constellation analysis (*sca*) model used for comparing monitored constellations with the expected one in terms of supervised features Y; *iii*) an AE-based (*ae*) model used for compressing and analyzing constellations using unsupervised features Z; and *iv*) the path metric estimation (*pme*) model used to predict the path length as a function of supervised features.

After some initializations, the lightpath model L is built by selecting, for each of the components in R, the available component model in *DB* that better fits component's attributes (lines 1-4 of Algorithm 1). After saving model L for further purposes (line 5), a number of random constellation samples are generated, propagated through L (see Fig. 5b) and saved in a temporary data set Y_{hist} (lines 6-13). Y_{hist} is processed in order to build the *sca* model. To this aim, the difference between the features of every pair $<Y_i, Y_j> \in Y_{hist}$ is computed by means of a statistical test based on the chi square test [20]

(lines 14-15). The proposed chi2-based statistic (*chi2*) is formally defined as:

$$chi2(Y_i, Y_j) = \sum_{k=1..|Y_i|} \frac{(Y_i(k) - Y_j(k))^2}{min(Y_i(k), Y_j(k))} \quad (1)$$

In consequence, the *sca* model includes the pair of reference samples $<Y_i, Y_j>$ that maximizes the value of *chi2*, as well as such maximum *chi2* value that will be later used as threshold for acceptable difference between samples (lines 16-18). Finally, *ae* and *pme* models and parameter $\mathcal{L}_{thr}$, are retrieved from the set of trained models and eventually returned jointly with *sca* (lines 19-22). Although *pme* can be designed in multiple ways, we consider it as a DNN model that predicts the lightpath length as a function of features *Y*.

B. Lightpath Length Analysis

Once the sandbox manager feeds the Rx agent with updated models, in-operation constellation analysis can be carried out.

Algorithm 2 details the procedure used to detect mismatch between the received and the expected constellation in terms of the supervised features *Y*. Therefore, it requires to process the monitored constellation *X* jointly with models *sca* and *pme*, and parameter $\mathcal{L}_{thr}$. After some initializations (line 1 in Algorithm 2), the supervised features are computed and the logarithm of the likelihood is compared against $\mathcal{L}_{thr}$ to detect whether observed features are unlike to follow a Gaussian distribution (lines 2-4). If so, a new message with the obtained likelihood is added to the diagnosis report (line 5).

The analysis continues by computing the *chi2* test between the observed features and the two reference samples stored in *sca*. The minimum of both values is compared against the threshold (lines 6-8), and, in case of threshold violation, an unexpected lightpath length is detected and *pme* is used to provide an estimation of the real length of the lightpath, which adds relevant information to the diagnosis report (lines 9-10). The procedure finishes by returning the diagnosis report, as well as the supervised feature sample *Y*, which can be eventually stored for further analysis (line 11).

C. Power Anomaly Detection

Finally, Algorithm 3 performs the analysis of the monitored constellation using the *ae* model. This analysis computes the relevance (importance) *h* of the *ae* model inputs and keeps track of them in time to detect any strong variation, like drift or shift. To that end, besides the monitored sample *X* and *ae* model, the algorithm receives the set of historical relevance measurements *H* of such a lightpath. First of all, forward analysis is performed (lines 1-6 in Algorithm 3). Specifically, the original sample *X* is transformed into the latent sample *Z* using the encoder and reconstructed into *X** using the decoder. Then, the rMSE between original and reconstructed sample is compared against the maximum error computed during *ae* training to detect whether latent features *Z* are inaccurate. Similarly as for the previous use case, inaccurate features are diagnosed if reconstruction error is high.

Regardless of the result of forward diagnosis, analysis

Algorithm 2. [Rx] - Length Analysis

INPUT: *X*, *sca*, *pme*, $\mathcal{L}_{thr}$; **OUTPUT:** *Y*, *diagnosis*

1: *diagnosis* ← ∅
2: *Y* ← GMMfitting(*X*)
3: $\mathcal{L}$ ← logLikelihood(*X*, *Y*)
4: **if** $\mathcal{L} > \mathcal{L}_{thr}$ **then**
5: *diagnosis*.add(<'Inaccurate Features', $\mathcal{L}$>)
6: $test_i$←computeChi2(*sca*.Y_i, *Y*) (eq. (1))
7: $test_j$←computeChi2(*sca*.Yj, *Y*) (eq. (1))
8: **if** min($test_i$, $test_j$) < *sca*.thr **then**
9: *diagnosis*.add(<'Unexpected Length, *test*>)
10: *diagnosis*.add(<'Estimated Length, *pme*(*Y*)>)
11: **return** *Y*, *diagnosis*

Algorithm 3. [Rx] - Importance analysis

INPUT: *X*, *H, ae*; **OUTPUT:** *Z*, *H*, *diagnosis*

1: *diagnosis* ← ∅
2: *Z* ← *ae*.encoder.propagate(*X*)
3: *X**←*ae*.decoder.propagate(Z)
4: ε←rMSE(*X*,*X**)
5: **if** ε > *ae*.ε_{max} **then**
6: *diagnosis*.add(< 'Inaccurate Features', ε>)
7: *h*←relevanceBackpropagation(*ae*, *X**)
8: *H*←update(*H*,*h*)
9: **if** detectVariation(*H*) **then**
10: *diagnosis*.add(< 'Power anomaly', *h*>)
11: **return** *Z*, *H, diagnosis*

continues by applying relevance backpropagation techniques [22], using the *ae* model backwards (from *X** to *X*) in order to compute the relevance of every input. By averaging the relevance of the inputs of the same constellation points, the relevance analysis vector *h* is computed, with one value for each constellation point in *P* (line 7).

Once the relevance vector *h* is computed, historical set *H* is updated (line 8). Note that *H* consists in |*P*| time series with the temporal evolution of the relevance of each constellation point. Then, a procedure to detect variations in time series can be applied, e.g., the procedure presented in [23] to detect variations such as gradual drift and instantaneous shift. This variation analysis can either be applied to each of the time series independently or the aggregation (sum) of time series belonging to a group of constellation points, e.g., outer or inner constellation points. If, regardless of the type, some variation is detected, a power anomaly message is generated and added to the report jointly with the current measured relevance (lines 9-10). Finally, the diagnosis report and the historical relevance set *H* are returned, jointly with the latent sample *Z* (line 11).

5. ILLUSTRATIVE NUMERICAL RESULTS

In this section, we first introduce the simulation scenario and constellation data generated for numerical evaluation purposes. Then, the feature extraction procedure presented in Section 3.A is evaluated and next, the DNN-based concatenated model for lightpath modelling, presented in Section 3.B, as well as the setup algorithm in Section 4.A are validated. Finally, the lightpath analysis use cases introduced in Sections 4.B and 4.C are evaluated.

A. *Simulation Scenario and Data Sets*

To evaluate the proposed methods for constellation and lightpath analysis, a MATLAB-based simulator of a coherent WDM system was developed to generate IQ constellations for a 16QAM@64GBd signal under different physical path characteristics. Assuming 100 GHz channel spacing and full spectrum occupancy, signal samples containing 2,048 symbols and shaped by a root-raised cosine filter with a 0.06 roll-off-factor are generated at the Tx side. Then, the signal is propagated through standard single mode fiber spans, characterized by optimal power of -1 dBm, attenuation factor of 0.21 dB/km, dispersion parameter of 16.8 ps/nm/km, and nonlinear parameter of 1.14 1/W/km. Spans are modeled by solving the nonlinear Schrödinger equation using the well-known split-step Fourier method, whereas ideal inline optical amplification is modelled as erbium-doped fiber amplifiers with a noise figure of 4.5 dB, introducing linear noise. Finally, a DSP block is considered at the Rx able to perform ideal chromatic dispersion compensation and phase recovery.

Fig. 6 shows the two different scenarios configured for constellation data generation. Under the *single link* scenario (Fig. 6a), a sequence of spans between Tx and Rx without intermediate ROADMs is configured. The first span has a variable length ranging from 40km to 80km and places an optical attenuator after the Tx to adjust the signal power according to the length of the first span, whereas the remaining spans have a fixed length of 80km. We considered up to 25 spans, so 3,000 constellation samples with a total length ranging from 80km to 2,000km were generated. Moreover, each sample belongs to one of the following configurations for the first span length and initial attenuation: *i*) *optimal* (80 km, 0 dB); *ii*) *sub-optimal* (60 km, -4 dB), *iii*) *degradation* (40 km, -8 dB). These different configurations have been devised to introduce power variations that result in small changes in the optical constellations without impacting lightpaths' QoT. In contrast, intermediate ROADMs between Tx and Rx are considered under the *multiple link* scenario (Fig. 6b). The WSSs inside the ROADMs are based on commercially available ones and modelled as described in [24]. In this case, four different optical link configurations in terms of total length and number of spans are considered: 100-km (2x50-km spans), 240-km (4x60-km spans), 400-km (5x80-km spans), and 560-km (7x80-km spans). By generating lightpaths with hop length (number of links in the lightpath's route) in [1, 4] and combining links with different

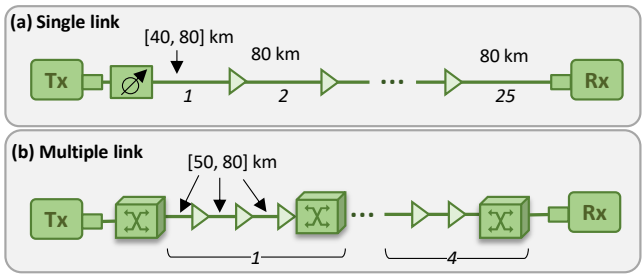

Fig. 6. Single link (a) and multiple link (b) lightpath scenarios.

configurations, 3,000 signals with a total length between 100km and 2,240km were generated. All generated data are openly available in [25]. In this work, we use a typical data split of 60%-20%-20% for training, testing, and validation purposes, respectively.

The network architecture in Fig. 1 has been reproduced in a Python-based simulator implementing its main elements and functional blocks. For model training in the sandbox domain, we used *sklearn* and *keras* as main libraries for training and testing DNN models. The configuration of each DNN model and the selected data set for training, testing, and validation is specified in the following subsections.

B. *Feature Extraction Evaluation*

Let us first focus on evaluating the supervised feature extraction methodology based on GMM fitting. For this study, we used the data generated under the single link scenario with the optimal power configuration. Fig. 7 shows the evolution of the supervised features as a function of the total lightpath length. For the sake of simplicity, only one outer (-3+3i) and one inner (-1-1i) constellation points are selected. We observe that the average position of constellation points (Fig. 7a), which has been normalized to the expected centroid, slightly varies with length. Nonetheless, the selected outer constellation point shows some remarkable drift in the Q axis. Regarding variance terms (Fig. 7b), it is clear that they are strongly correlated with length in the whole range, whereas the covariance term (Fig. 7c) has a significant shift for long path lengths. We observe in the figures that clear and strong patterns between the supervised features and the length of the lightpath exist, which anticipates good accuracy of the length analysis procedures based on these supervised features.

In addition to the previous results, Fig. 7d shows the likelihood GoF metric $\mathcal{L}$, which stays above -3.5 for all the

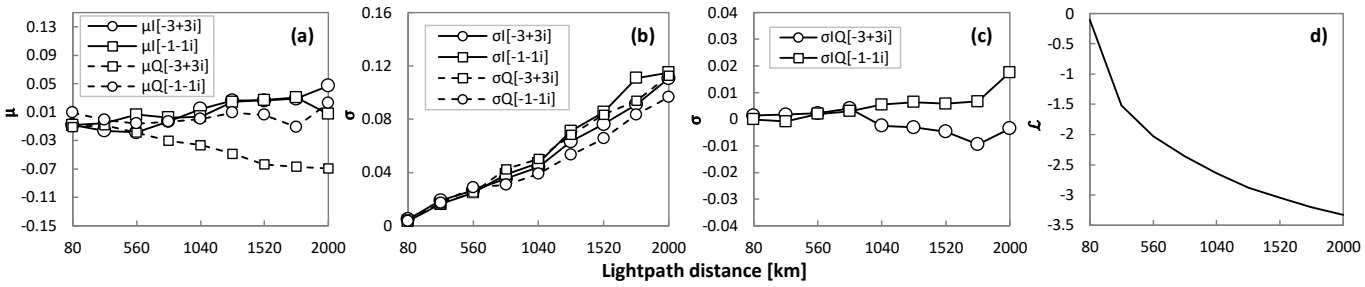

Fig. 7. Supervised feature extraction performance.

considered distances. Then, such value can be selected as $\mathcal{L}_{thr}$ parameter for validation of feature extraction procedures. Moreover, the Henze-Zirkler multivariate normality test [26] was conducted for all the samples belonging to the selected dataset. We concluded that all 16 constellation points can be accurately modelled as Gaussian distributions for all the considered distances, since the obtained p-value of the test always exceeded the commonly accepted significance level of 0.05. To better illustrate the valid fitting of constellation points as bi- variate Gaussian distributions, Fig. 8 zooms in the selected inner and outer constellation points of two samples belonging to different lightpath lengths (400km and 1600km). The computed Gaussian distribution is plotted together with the samples, showing different level curves for different variance values. In view of the results, we can validate the proposed supervised feature extraction procedure for the characterization of constellation points.

Let us now numerically evaluate the performance of the unsupervised feature extraction based on the AE model, which is part of the forward analysis. To this aim, we initially trained an AE model with data from the single link scenario and optimal power configurations. Fixing a symmetric encoder and decoder configuration, each with 4 hidden layers (1024, 256, 128, and 64 ReLU neurons), we trained different AEs for a size of latent space Z (m) ranging from 4 to 64 features. The results in terms of reconstruction error ε for the testing samples are presented in Fig. 9a. For benchmarking purposes, Principal Component Analysis (PCA) [27] was conducted, where the training data set was used to compute the first m PCs that collect the maximum information from the original data. Testing data samples were next compressed and reconstructed with the selected PCs, thus emulating the encoding/decoding AE network components. Supported by the results, we can conclude that 32 latent features are enough to reach a negligible (< 2%) reconstruction error. Note that the same number of PCs doubles the error of that of the AEs.

Fig. 9b plots the compression rate achieved as a function of the target reconstruction fidelity (defined as 1- ε). We observe that the AE clearly outperforms PCA. Note that 99% of reconstruction fidelity can be achieved with compressed samples, while reducing in 96% the size of original constellation samples. Such an extremely large compression rate is not achieved by PCA, which reaches a moderated 60% of compression rate for the same reconstruction fidelity.

Finally, Fig. 9c plots the relation between unsupervised features and path length, similarly as for supervised features. For representation purposes, the 32 latent features have been projected into two dimensions by means of applying PCA to the latent space samples. The graph shows the position of the samples in the two-dimensional space obtained with PCA, whereas path length is coded by a color scale. We observe a clear relation between latent features and path length. However, this relation is not as strong as that of the supervised features, which validates the latter for path length analysis.

C. *Lightpath Modelling*

For lightpath modelling, we selected the number of constellation points to the minimum providing just enough information to capture the overall constellation characteristics; for 16QAM, specifically, two outer (-3+3i, 1-3i) and two inner (1+1i and -1-1i) constellation points were selected. In addition, we considered that both DNN models for the optical fiber links and for ROADMs follow the same architecture characterized by: *i*) 20 input neurons (5 features per constellation point); *ii*) two hidden layers, each one with 12 neurons and *tanh* activation function; and *iii*) one output

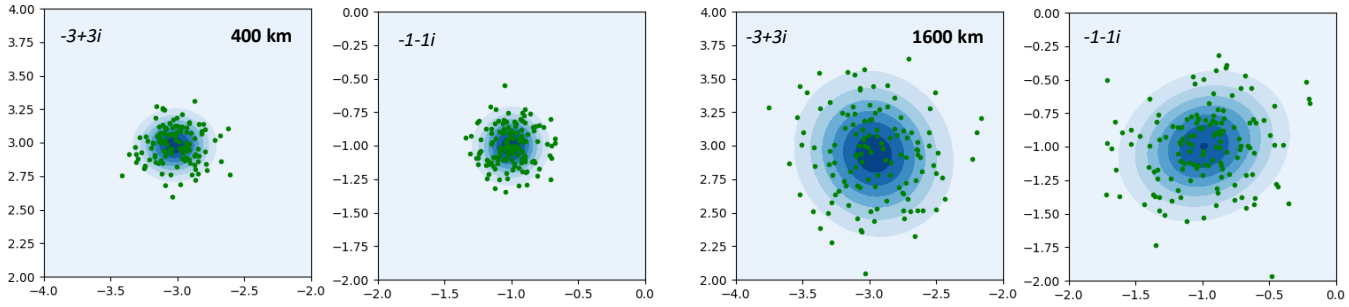

Fig. 8. Example of supervised feature extraction for two constellation points after 400km and 1,600 Km.

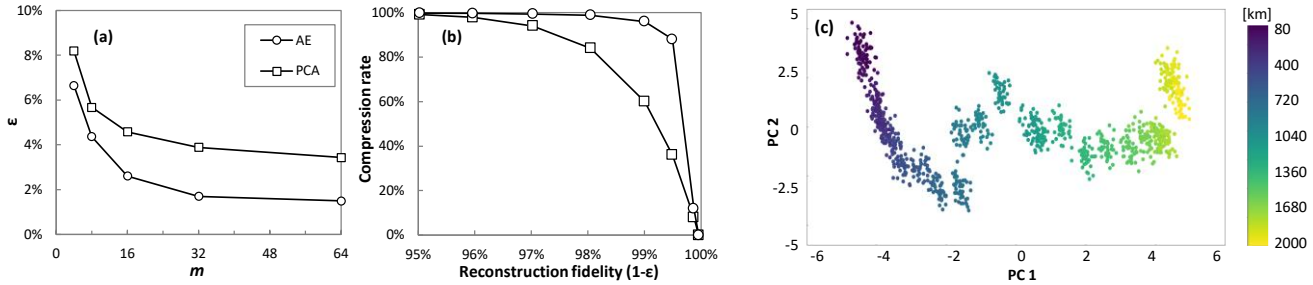

Fig. 9. Unsupervised feature extraction performance.

layer with 24 neurons to estimate the output features. These component models were trained during 5,000 epochs and tested with data from the multiple link data set.

The overall absolute and relative errors for all link configurations, lightpath lengths, and selected constellation points are shown in Fig. 10, where average and maximum errors of features μ (Fig. 10a) and σ (Fig. 10b) are plotted. We observe negligible μ prediction errors (max error < 2%) independently of the link length. In contrast, σ max error is around 30% for low σ values although decreases when path length increases, becoming under 15%, which is, in general, a good enough performance to validate the models. For illustrative purposes, Fig. 10c plots the Gaussian distributions for the selected constellation points, obtained with the concatenation model and simulator for a 1,600-km lightpath (4 optical links of 400km). For the sake of clarity, we reduced the number of level curves and removed colors. It is worth noting that strong similarities between both cases are evident.

The reconstruction of the features of the non-selected constellation points can be carried out by means of the proposed linear model with a reconstruction accuracy of 97%, which indicates the proper choice of the selected points.

D. *Length Analysis Use Case*

Let us now analyze the performance of Algorithm 2. Specifically, we focus on the performance of models *sca* and *pme*. To this aim, the features of the reference samples stored in *sca* model (*model-based*) are compared to those extracted from the validation samples of the multiple link data set (*simulator-based*). The comparison between model-based and simulator-based features was performed using the proposed *chi2* test. We first compared the case when the simulation and model were configured with the same lightpath length and link configuration. For all the combinations, the maximum observed threshold for the *chi2* test never exceeded 0.5 in logarithm scale. Therefore, we use such a threshold for unexpected length detection.

Next, we compared different configurations of 4-link hop lightpaths to check whether the value of the *chi2* test serves as a good indicator of misleading length. Fig. 11a reports the results, where we observe that the selected threshold of 0.5 allows us to clearly distinguish all cases when simulation and model were configured differently (orange) from cases with the same length (green). Additionally, the impact of considering just slightly different scenarios in the simulation and concatenation model was tested. Specifically, a 4-link hop lightpath with 240-km links was configured in the simulation, whereas the model was configured with the same number of hops and link configuration except for only one of the hops, where a 400-km link was selected. The four different positions in the path for the 400-km link were evaluated; Fig. 11b shows that all cases stayed above the 0.5 threshold, which implies that the small difference was correctly detected. Note that localization of the longer link can be done by performing the test in the intermediate links. Finally, Fig. 11c shows the result of applying the intermediate analysis when the 400 km is in the third fiber span. We observe that the link is localized as the *chi2* test value exceeds the selected threshold when evaluating the features right after the third fiber span.

The performance of *pme* was eventually evaluated. The structure of *pme* DNN was as follows: *i*) 20 input neurons (5

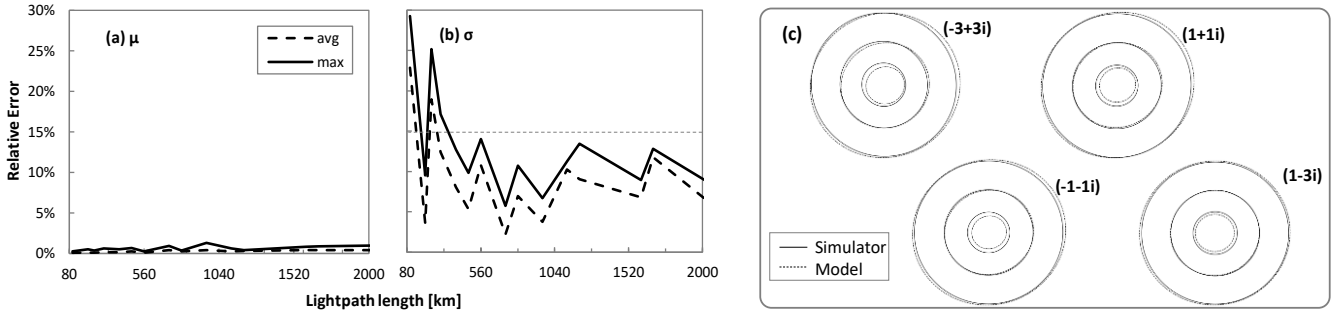

Fig. 10. Lightpath modelling performance.

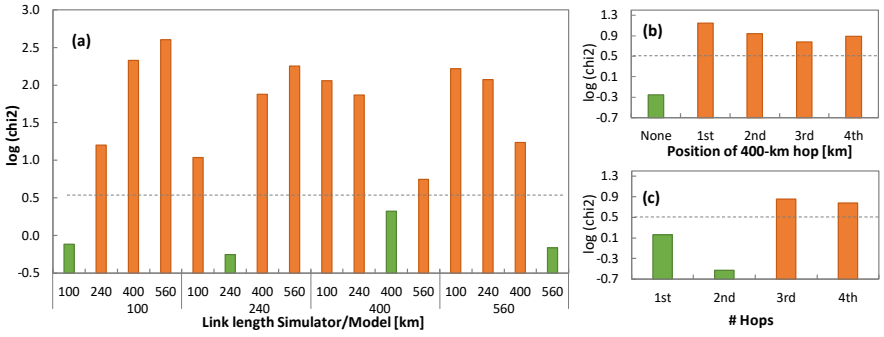

Fig. 11. *sca* model performance.

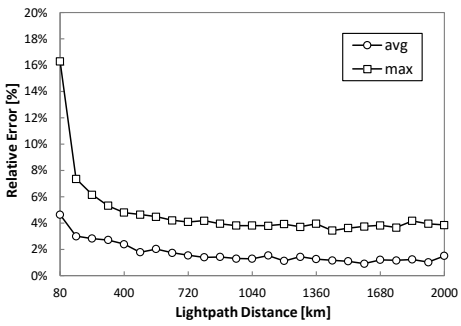

Fig. 12. *pme* model performance.

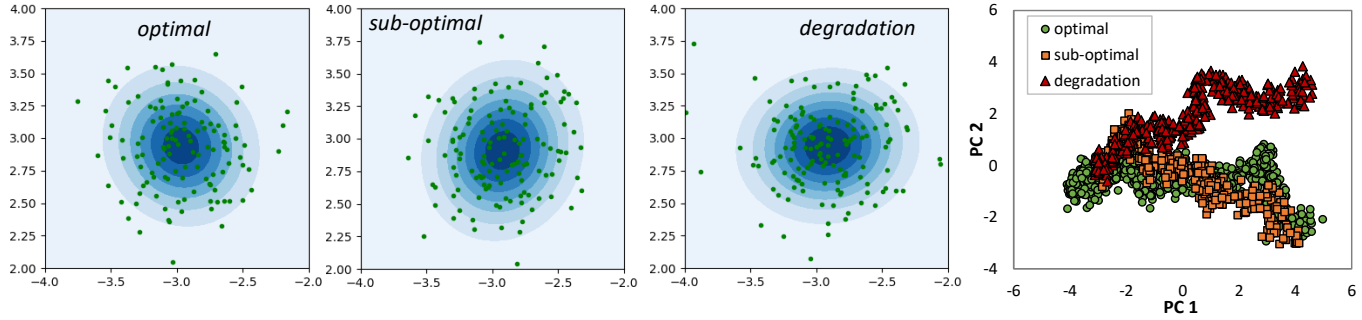

Fig. 13. Constellation point -3+3i examples for power scenarios.

Fig. 14. Power scenario discrimination with *Y*.

features per selected constellation point); *ii*) two hidden layers, each one with 12 neurons and *tanh* activation function; and *iii*) one output layer with one single neuron that predicts lightpath length. The *pme* DNN model was trained with the single link data set and during 5000 epochs. Fig. 12 shows the average and maximum relative estimation error. Here, we observe average error below 5% regardless of lightpath length and maximum error under 10% for lightpaths longer than 100km. These results validate *pme* as accurate real lightpath length estimation.

E. Relevance Analysis

Finally, we conducted a numerical study to evaluate the models involved in Algorithm 3. Specifically, we focused on illustrating how the *ae* model can discern between different power scenarios and on showing how the input relevance varies with power degradation.

For power scenario discrimination, let us first inspect the examples of constellation point -3+3i for each scenario reproduced in Fig. 13. We observe clear differences on the Gaussian features *Y* among power scenarios. Fig. 14 shows each of the samples projected on the reduced two-dimensional PCA space from the *Y* space. We realize that the observed differences do not support an easy discrimination since the three classes are not separable. Nonetheless, as part of the forward analysis, the same projection can be performed using latent space samples *Z* computed with the *ae* model trained with all power configurations. This AE produces a reconstruction error under 2%, similar to the one in Section 5.B for unsupervised feature extraction. Fig. 15 presents the obtained results where the three power scenarios are clearly distinguishable. In consequence, we conclude that the proposed AE-based model for constellation analysis allows for accurate discrimination of power configurations producing small changes in the received constellation.

Regarding relevance analysis, let us compare two different ways to aggregate relevance of constellation points: *i*) quadrant-based, e.g., right-upper and left-bottom quadrants (Fig. 16a); and *ii*) energy-wise, i.e., inner and outer constellation points (Fig. 16b). In view of the figures, we conclude that energy-wise aggregation gives more information, since the relevance of inner constellation points clearly reduces when power degrades. Hence, relevance

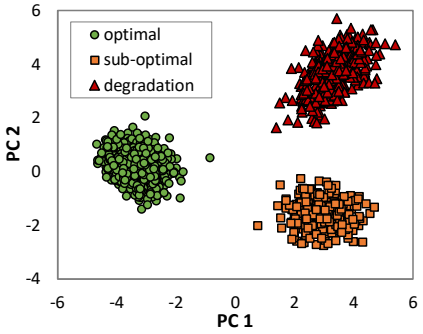

Fig. 15. Power scenario discrimination with *Z*.

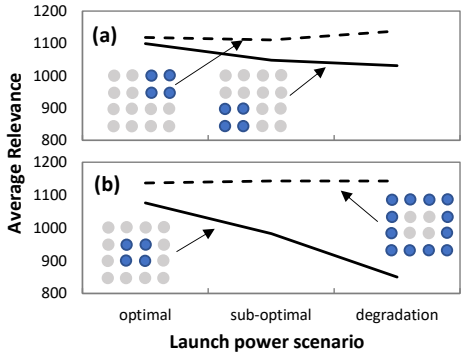

Fig. 16. Relevance vs power scenarios.

Table 1. Input relevance variation due to power degradation.

I/Q	-3	-1	1	3
3	7%	-2%	-4%	-9%
1	-4%	-17%	-12%	13%
-1	-3%	-9%	14%	-4%
-3	10%	5%	-3%	22%

analysis can be potentially used to early power anomalies detection, which could eventually lead to hard failures.

The relative relevance variation per constellation point when moving from optimal to suboptimal scenario is detailed in Table 1. Increase/decrease of relevance in more than 5% above/below the reference optimal configuration is highlighted in green/red. In line with the conclusions from Fig. 16, we can easily verify that outer constellation points (mainly those in the corners) become more relevant, since

more NLI and LI noise is expected under sub-optimal power configuration, which makes the shape of outer points more elliptical than those with lower energy. Hence, the importance of the symbols on these outer constellation points in the latent space is higher, since the overall shape of the constellation is more complex.

6. CONCLUDING REMARKS

A comprehensive DL-based IQ constellations analysis for in-operation lightpath modelling and power analysis has been proposed. DL models propagating IQ constellations features were trained in a sandbox domain for modelling optical components, such as optical links, OAs, and ROADMs. Then, a target lightpath can be modelled by concatenating specific DL models, to reproduce the propagation of IQ constellations from Tx to Rx. Two methods for feature extraction were proposed, based on GMM (supervised) and on AE (unsupervised). By using those models at the node agent, real-time analysis of the received optical signal can be carried out. In addition, constellation samples were compressed into a reduced set of latent space features, which remarkably reduces (more than 95%) the amount of data that needs to be sent to the centralized controller.

Two illustrative use cases of lightpath performance analysis were investigated. First, lightpath length analysis showed noticeable low error for lightpaths longer than 100km, clearly detecting slight differences in lightpath configurations. Next, different power profiles were studied, where extracted latent features from AE models showed accurate discrimination in terms of Tx power configurations. Finally, a relevance constellation analysis for the AE input parameters was carried out, providing clear understanding about which constellation points have larger relevance, which might be very useful for different analysis proposes.

Funding. The research leading to these results has received funding from the H2020 B5G-OPEN (G.A. 101016663), the MSCA REAL-NET project (G.A. 813144), the MICINN IBON (PID2020-114135RB-I00) project, and the ICREA Institution.

REFERENCES

1. H. Hallingby, S. Fletcher, V. Frascolla, A. Gavras, I. Mesogiti, and F. Parzysz, "5G Ecosystems," *white paper*, 5G-IA, 2021. [On-line] https://doi.org/10.5281/zenodo.5094340.
2. L. Velasco, M. Signorelli, O. González, C. Papagianni, R. Bifulco, J. Vegas Olmos, S. Pryor, G. Carrozzo, J. Schulz, M. Bennis, R. Martinez, F. Cugini, C. Salvadori, V. Lefebvre, L. Valcarenghi, and M. Ruiz, "End-to-End Intent-Based Networking," IEEE Communications Magazine, vol. 59, pp. 106-112, 2021.
3. B. Shariati, L. Velasco, J. Pedreno, A. Dochhan, R. Casellas, A. Muqaddas, O. González, L. Canto, B. Lent, J. López, S. López, F. Moreno, P. Pavón, M. Ruiz, S. Patri, A. Giorgetti, F. Cugini, A. Sgambelluri, R. Nejabati, D. Simeonidou, R. Braun, A. Autenrieth, J. Elbers, J. Fischer, R. Freund, "Demonstration of Latency-Aware 5G Network Slicing over Edge Computing Enabled Optical Metro Network," IEEE/OSA J. of Optical Comm. and Networking, vol. 14, pp. A81-A90, 2022.
4. L. Velasco, A. Chiadò, O. González, A. Lord, A. Napoli, P. Layec, D. Rafique, A. D'Errico, D. King, M. Ruiz, F. Cugini, R. Casellas, "Monitoring and data analytics for optical networking: benefits, architectures, and use cases," IEEE Network Magazine, vol. 33, pp. 100-108, 2019.
5. D. Rafique and L. Velasco, "Machine Learning for Optical Network Automation: Overview, Architecture and Applications," IEEE/OSA J. of Optical Comm and Networking, vol. 10, pp. D126-D143, 2018.
6. W. Saif, M. Esmail, A. Ragheb, T. Alshawi and S. Alshebeili, "Machine Learning Techniques for Optical Performance Monitoring and Modulation Format Identification: A Survey," IEEE Communications Surveys & Tutorials, vol. 22, pp. 2839-2882, 2020.
7. Q. Zhuge, X. Zeng, H. Lun, M. Cai, X. Liu, L. Yi, W. Hu, "Application of machine learning in fiber nonlinearity modeling and monitoring for elastic optical networks," IEEE/OSA Journal of Lightwave Technology, vol. 37, pp. 3055-3063, 2019.
8. F. Vaquero, D. Ives, C. Laperle, D. Charlton, Q. Zhuge, M. O'Sullivan, and S. Savory, "Machine Learning Based Linear and Nonlinear Noise Estimation," IEEE/OSA J. of Optical Comm and Networking, vol. 10, pp. D42-D51, 2018.
9. D. Wang, M. Zhang, J. Li, Z. Li, J. Li, C. Song, and X. Chen, "Intelligent constellation diagram analyzer using convolutional neural network-based deep learning," Optics Express, vol. 25, pp. 17150-17166, 2017.
10. VPI Photonics Design Suite. [On-line] https:// www.vpiphotonics.com.
11. M. Filer, M. Cantono, A. Ferrari, G. Grammel, G. Galimberti, V. Curri, "Multi-Vendor Experimental Validation of an Open Source QoT Estimator for Optical Networks," IEEE/OSA J. of Lightwave Technology, vol. 36, pp. 3073-3082, 2018.
12. B. Shariati, M. Ruiz, J. Comellas, L. Velasco, "Learning from the Optical Spectrum: Failure Detection and Identification [Invited]," IEEE/OSA J. of Lightwave Technology, vol. 37, pp. 433-440, 2019.
13. S. Barzegar, M. Ruiz, A. Sgambelluri, F. Cugini, A. Napoli, L. Velasco, "Soft-Failure Detection, Localization, Identification, and Severity Prediction by Estimating QoT Model Input Parameters," IEEE Trans. on Network and Service Management, vol. 18, pp. 2627-2640, 2021.
14. J. Song, C. Häger, J. Schröder, A. Graell, H. Wymeersch, "Model-Based End-to-End Learning for WDM Systems with Transceiver Hardware Impairments," arXiv preprint arXiv:2111.14515, 2021.
15. Q. Meng, D. Catchpoole, D. Skillicorn, P. Kennedy, "Relational autoencoder for feature extraction," in Proc. International Joint Conference on Neural Networks, 2017.
16. M. Ruiz, J. Morales, D. Sequeira, and L. Velasco, "An Autoencoder-Based Solution for IQ Constellation Analysis," in Proc. European Conference on Optical Communication (ECOC), 2021.
17. N. Bouguila and W. Fao, *Mixture Models and Applications*, Springer International Publishing, 2020.
18. L. Velasco, B. Shariati, F. Boitier, P. Layec, M. Ruiz, "A Learning Life-Cycle to Speed-up Autonomic Optical Transmission and Networking Adoption," IEEE/OSA J. of Optical Comm. and Networking, vol. 11, pp. 226-237, 2019.
19. M. Dallaglio, A. Giorgetti, N. Sambo, L. Velasco, and P. Castoldi, "Routing, Spectrum, and Transponder Assignment (RSTA) in Elastic Optical Networks," IEEE/OSA J. of Lightwave Technology, vol. 33, pp. 4648-4658, 2015.
20. R. Kuehl, *Design of Experiments: Statistical Principles of Research Design and Analysis*, 2nd Ed., Duxbury Press, 1999.
21. C. Aggarwal, *Neural Networks and Deep Learning: A Textbook*, Springer, 2018.
22. A. Binder, S. Bach, G. Montavon, K. Müller, and W. Samek, "Layer-wise Relevance Propagation for Deep Neural Network Architectures," in Proc. Information Science and Applications, pp. 913-922, 2016.

23. 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.
24. T. Rahman, A. Napoli, D. Rafique, B. Spinnler, M. Kuschnerov, I. Lobato, B. Clouet, M. Bohn, and Ch. Okonkwo, “On the Mitigation of Optical Filtering Penalties Originating from ROADM Cascade,” IEEE Photonics Techn. Letters, vol. 26, pp. 154-157, 2014.
25. M. Ruiz, L. Velasco, D. Sequeira “Replication Data for: Optical Constellation Analysis (OCATA)", https://doi.org/10.34810/data146, V1, 2021.
26. B. Ebner, and N. Henze, “Tests for multivariate normality—a critical review with emphasis on weighted L2-statistics [invited],” Springer TEST, vol. 29, pp. 845-892, 2020.
27. E. Barshan, A. Ghodsi, Z. Azimifar, and M. Jahromi, “Supervised principal component analysis: visualization, classification and regression on subspaces and submanifolds,” Elsevier Pattern Recognition, vol. 44, pp. 1357-1371, 2011.