A Comparative Study on Principal Components Analysis for Failure Detection in Optical Networks

Ingrid Ramos, Adryele Oliveira, Andrei Ribeiro, Rafael Sales, Fabrício Lobato, João C.W.A. Costa, and Moisés Silva.

***Abstract*— The application of failure detection techniques in transparent optical networks is crucial to ensure the reliability of these networks and prevent data losses. Hence, this work compares the failure management performances of two unsupervised learning algorithms based on Principal Component Analysis (PCA): the linear PCA and the non-linear PCA (NLPCA) built from auto-associative neural networks. These techniques may be trained with only data from normal conditions, handling the imbalanced nature of the dataset. Experimental results acquired with a testbed-derived dataset show that the two PCA-based techniques detect and locate failures with over 90% accuracy.**

***Keywords*— Optical Networks, Machine Learning, Principal Components, Failure Detection.**

I. Introduction

Leveraging the accelerated spread of large bandwidth networks that came with the development of 5G, the possibility of having applications operating with high data transmission rates has arisen, as well as the improvement of technologies such as the Internet of Things, cloud computing, and autonomous vehicles. Since transparent optical networks (TONs) are promising technologies capable of dealing with this massive data flow, it is important to guarantee their quality of transmission (QoT). In that scenario, promoting accurate and real-time failure management is paramount to ensure the reliability of TONs.

Typically, the most traditional methods for failure management in optical networks are based on predefined thresholds or probability statistics models [1]. However, current TONs are becoming more flexible, self-adaptive, and dynamic. Therefore, threshold-based methods might be inefficient in properly identifying a failure, which leads to more cost and a higher probability of errors [2]. In that case, alternatives that contribute to the autonomy of networks and with adaptive thresholding become crucial.

To address this situation, the application of machine learning (ML) techniques for failure management has been explored, as they can learn patterns that differentiate between normal and faulty conditions regarding network components [3]. However, supervised learning (SL) models are the most often applied strategy for failure detection [4], which requires a balanced dataset (i.e. when the number of samples from both normal and faulty conditions is similar). However, in real scenarios, there are much fewer anomalous data (if any) than normal data, leading to poor training of SL models, filled with false alarms, and shortcomings in fault recognition.

Ingrid Ramos, Adryele Oliveira, Andrei Ribeiro, Rafael Sales, Fabrício Lobato, and João CWA Costa, Insitute of Technology, Federal University of Pará, Belém-Pará, Brazil, e-mail: [ingrid.ramos, adryele.oliveira, andrei.ribeiro, rafael.sales]@itec.ufpa.br, [frl, jweyl]@ufpa.br; Moisés F. Silva, Los Alamos National Laboratory, Los Alamos-NM, USA, e-mail: mfelipe@lanl.gov

Recent studies have proposed the usage of unsupervised learning (UL) techniques [5] [6]. Unlike SL models, these techniques only need data from normal conditions (i.e. without failures) to be properly trained. Therefore, acquiring data from equipment undergoing failures or the simulation of failure conditions from statistical models is unnecessary. More specifically, UL models can learn hidden patterns from normal data, recognizing underlying information. Thus, when anomalies (failures), i.e., data that have different behavior from normal conditions, are fed to the model, it can recognize them.

In that regard, in this work, we compare two UL techniques based on the Principal Components Analysis (PCA) algorithm for failure management in optical networks: the linear PCA and the non-linear PCA (NLPCA).

The rest of the paper is structured as follows. Section II is devoted to presenting the theoretical fundamentals of two PCA-based models. In Section III, we present the main operating principles of the PCA and NLPCA and how they can be implemented to perform failure detection and localization in the context of optical communication networks. Section IV describes the optical setup used for data acquisition and the optical dataset used in our experiments. Section V exhibits performance results and discusses some comparisons. Finally, in Section VI, we present our conclusions.

II. Approaches Based on Principal Component Analysis

A. Linear PCA

PCA is a widely used technique in ML and statistics for dimensionality reduction and data visualization [7]. PCA can reduce the data dimensionality by transforming it into a new coordinate system, where the variables are uncorrelated, orthogonal, and ordered by the amount of variance they capture. This transformation is achieved by finding the eigenvectors and eigenvalues of the covariance matrix.

In general, given a dataset X with n observations and p variables, the first step in PCA is to center the data. This is done by subtracting the mean of each variable from the respective variable values. This process ensures that the new coordinate system is aligned with the directions of maximum variance in the data rather than being centered on the means of the variables. Consequently, the arrangement of data points in this new coordinate system more accurately represents the underlying variance structure of the data. The centered data matrix is denoted as X'. Next, PCA computes the covariance

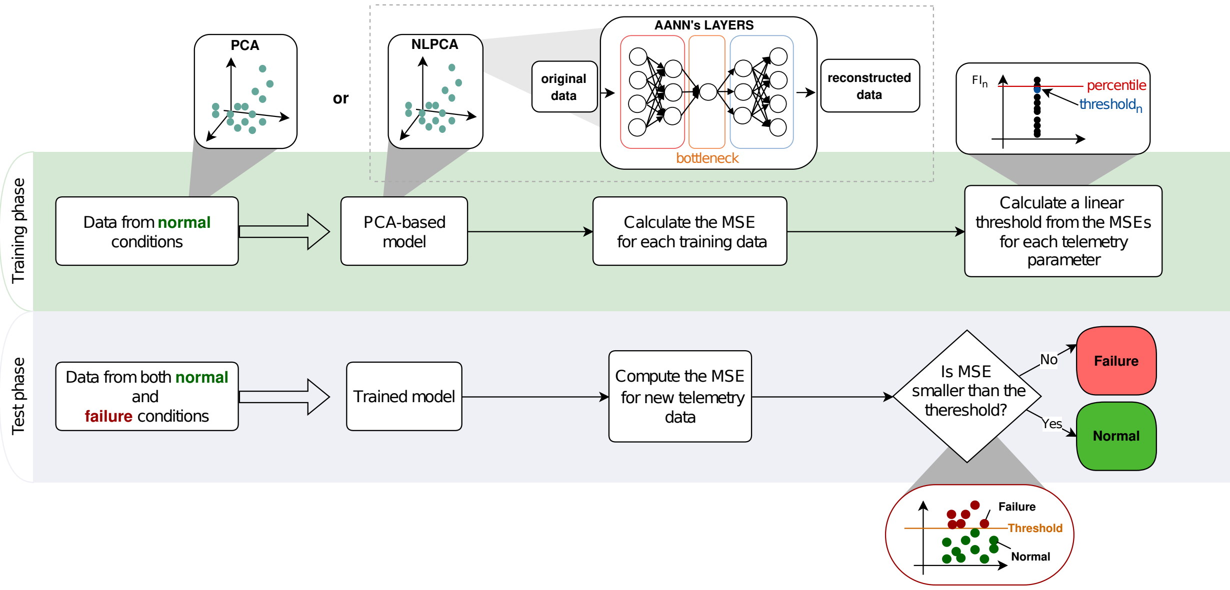


Fig. 1: Overview of the PCA-based failure management.

matrix of the centered data X'. Each element of this matrix is calculated as the covariance between pairs of variables, and the matrix is then rearranged into a square symmetric matrix. Besides, the eigenvectors decomposition is computed. The eigenvectors $\mathbf{v}$ of a covariance matrix $\mathbf{A}$ in PCA are found by solving the characteristic equation:

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}, \tag{1}$$

Where involves the following variables: $\mathbf{A}$, which represents the covariance matrix; $\mathbf{v}$, the eigenvector of the matrix $\mathbf{A}$; and λ, the eigenvalue corresponding to $\mathbf{v}$. To find the eigenvectors, one must compute the eigenvalues λ by solving the determinant equation:

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0. \tag{2}$$

Where $\mathbf{I}$ is the identity matrix. After finding the eigenvalues, the corresponding eigenvectors are estimated by substitution into the characteristic equation. PCA then calculates eigenvectors $\mathbf{v_1}$, $\mathbf{v_2}$, ... $\mathbf{v_p}$ and eigenvalues $\lambda_1, \lambda_2, ..., \lambda_p$ of $\mathbf{A}$. These eigenvectors form the new basis vectors of the transformed coordinate system.

Finally, PCA selects a subset of the eigenvectors, called principal components (PC), based on the amount of variance they capture. The k PCs corresponding to the k largest eigenvalues represent the most important directions of variation in the data. In other words, the covariance matrix represents the relationships between the variables in the dataset. The eigenvectors and eigenvalues provide important information about the directions of maximum variance and the amount of variance captured by each direction, respectively.

B. Non-linear PCA

NLPCA has the same objective as the traditional PCA. It groups correlated values from the dataset in a smaller dimension until obtaining k principal components [8]. Beyond this, it can be referred to as an extension of the traditional PCA method that allows for nonlinear mappings of features. Correspondingly, NLPCA aims to find a low-dimensional representation of high-dimensional data while preserving the inherent nonlinear data structure.

One common approach to NLPCA is using auto-associative neural networks (AANNs). These AANNs are feed-forward neural networks that, in addition to the input/output layers, comprise two mapping/de-mapping layers and one extra bottleneck layer to span the compressed data representation [9]. As shown in Fig.1, the bottleneck layer describes a code used to represent the input by performing a mapping $\boldsymbol{h} : \mathbb{R}^p \to \mathbb{R}^k$ of the input $\mathbf{x}$. Afterward, the data are reversed to the original space by a de-mapping operation $\boldsymbol{g} : \mathbb{R}^k \to \mathbb{R}^p$. The learning process aims to find the set of parameters $\Theta = \{\mathbf{W}, \mathbf{W}'\}$ that minimizes the loss function.

$$L(\Theta) = \frac{1}{n} \sum_{\forall \mathbf{x} \in \mathbf{X}} \|\mathbf{x} - \boldsymbol{g}_{\mathbf{W}'}(\boldsymbol{h}_{\mathbf{W}}(\mathbf{x}))\|^2, \tag{3}$$

where $L(\cdot)$ is a loss function penalizing $\boldsymbol{g}_{\mathbf{W}'}(\boldsymbol{h}_{\mathbf{W}}(\mathbf{x}))$ for being dissimilar from $\mathbf{x}$, i.e., a mean square reconstruction error (MSE). The most common approach for the encoder and decoder is through affine mappings together with nonlinear functions such as:

$$\begin{aligned} \boldsymbol{h}_{\mathbf{W}}(\mathbf{x}) &= s_h(\mathbf{W}\mathbf{x} + \mathbf{b}), \\ \boldsymbol{g}_{\mathbf{W}'}(\mathbf{x}) &= s_g(\mathbf{W}^T\mathbf{x} + \mathbf{a}). \end{aligned} \tag{4}$$

Thereby, the set of parameters turns into $\Theta = \{\mathbf{W}, \mathbf{b}, \mathbf{a}\}$, where $\mathbf{b}$ and $\mathbf{a}$ are the biases and $\mathbf{W}$ is the weight matrix. This scheme of shared weights is called symmetric architecture.

The proposed approach overview is shown in Fig. 1. Once the models are trained, they generate n MSE values for each of the p features in the dataset; these values can then be used for failure management. This process is repeated for each model since they are similar and can all produce MSE values.

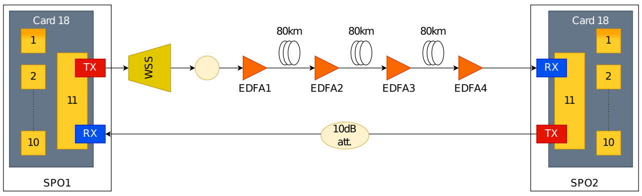


Fig. 2: Optical network testbed.

III. PCA-Based Approach For Failure Management

Firstly, only data from normal conditions (i.e. without failures) are fed into the models during the training phase. Then, they are trained to reconstruct this data in their output back to p dimensions after reducing to k PCs. The aim is to minimize the MSE, which measures how accurately the data was recreated. By learning hidden patterns from this data, the models picture the inherent characteristics of normal conditions.

At the end of training, MSEs are typically small, and any remaining variances are considered acceptable. This is defined by a set of linear thresholds automatically defined based on a chosen h percentile value; this process limits the n MSE values to the $h\%$ lowest for each of the p sorted vectors from the training dataset. The highest of these values becomes the threshold for each dimension, indicating the maximum MSE allowed for a sample to be considered normal. These thresholds are the core of failure management as they classify failures and non-failures.

Once the thresholds for each individual feature p are defined, the test phase, consisting of both normal and anomalous data, is carried out by computing the reconstruction errors and comparing them to the given threshold value for actual failure detection. Failure localization is naturally performed in this framework. A threshold value is automatically assigned to each feature (i.e., telemetry parameter). If the particular feature has an error value above that threshold, it is possible to detect and locate the presence of the failure.

IV. Results

A. Experimental Setup and Parameter Definition

To evaluate the proposed approach, the telemetry dataset described in [10] was used. The considered testbed includes two Ericcson SPO 1400 devices, one Wavelength Selective Switch (WSS), and four EDFA amplifiers. At the end of the WSS, an attenuator is installed to simulate failures, as shown in Fig. 2. The dataset is composed of 10 hours. In the first 8 hours, two normal operation conditions (i.e., without failures) were simulated: a stationary normal behavior during the first 6 hours and a noisy normal behavior in the remaining 2 hours by randomly changing the attenuation at the range from 0 to 18 dB. In the remaining 2 hours among the total 10 hours, the same behavior from the last 8 hours is simulated, but a 25dB attenuation is added every 40 seconds, putting the network in a failure condition for 10 seconds. After that, the WSS is reconfigured so that the network starts working properly again.

The optical connection comprised three 80 km spans between the SPO-TX and SPO-RX. The data is collected with a sampling frequency of 3.5 seconds and consists of 4 features corresponding to the 4 EDFAs input powers. An interpolation technique was employed due to missing values in the original data set, generating at the end 13,948 samples. Among the samples, the first 80% of data are used for training and the following 20% for testing. Notably, failure conditions were exclusively part of the test phase data. Fig. 3 presents the four features that compose the dataset and are split into training and testing sets.

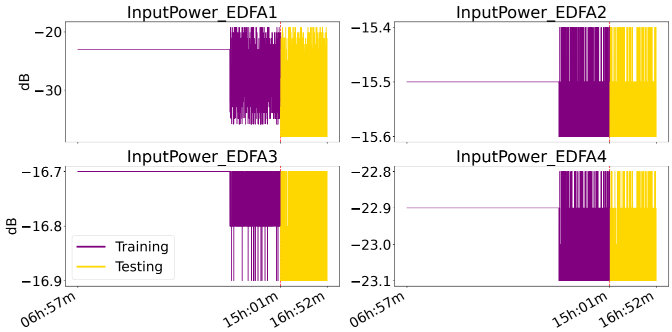


Fig. 3: Input Power on EDFAs over time.

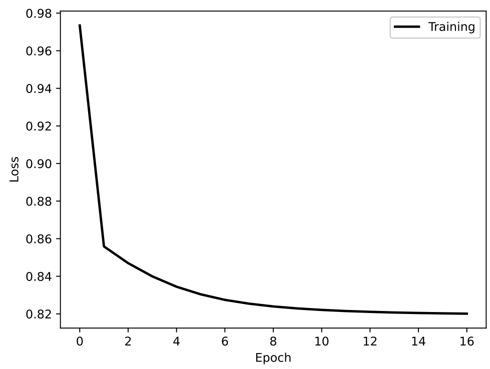


Fig. 4: NLPCA's loss function by epochs.

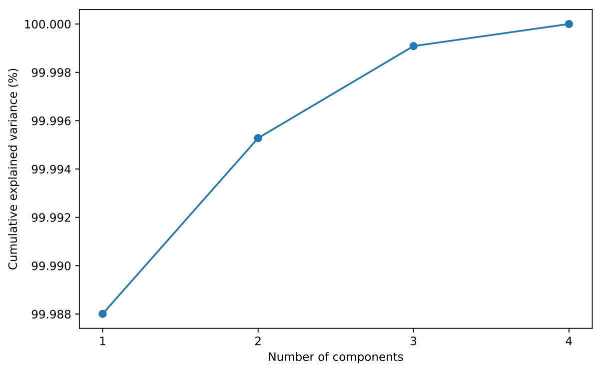
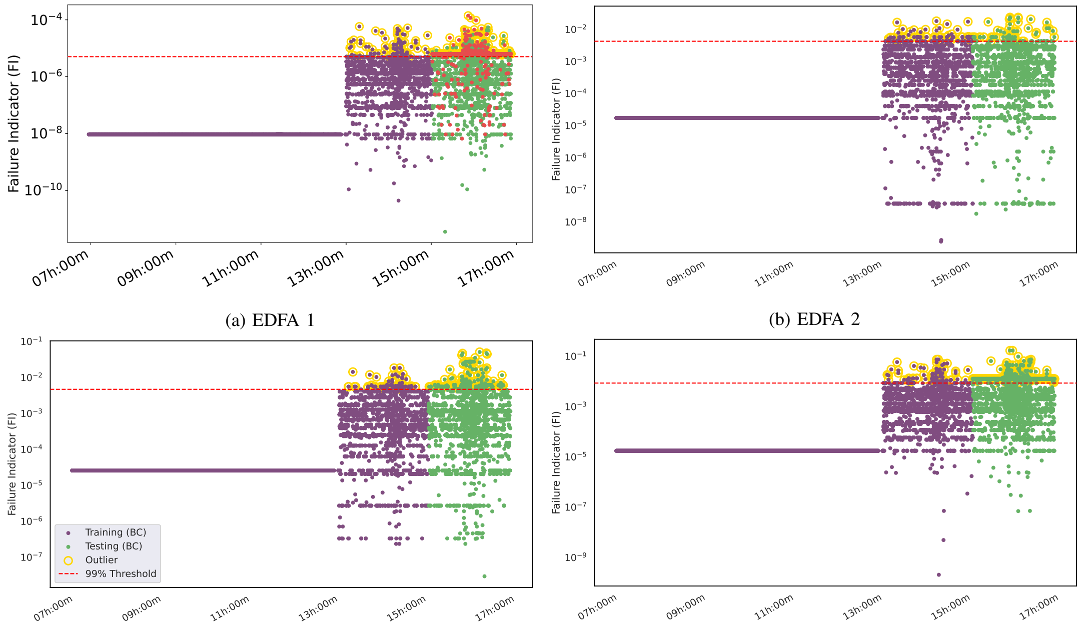


Fig. 5: Cumulative explained variance by number of components.

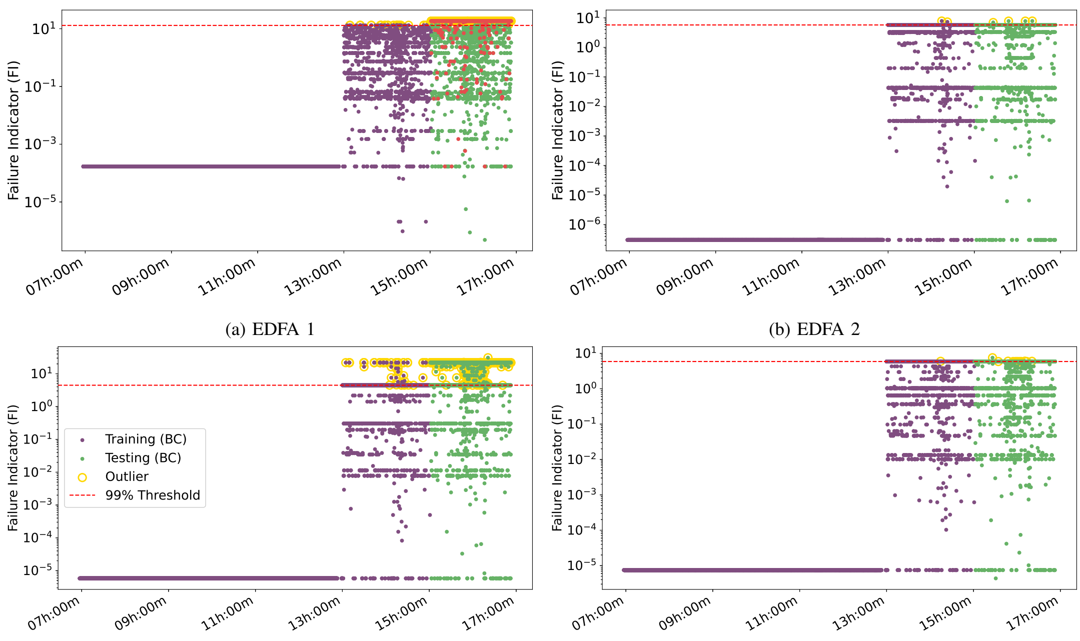
Specifically for NLPCA, the number of epochs must be defined preferentially when the loss between epochs is almost constant. In that case, it was set for when the retention of data variance is superior to 99.9%, which happens around the 12th period (for a batch size equal to 10), as shown by the function



(a) EDFA 1 (b) EDFA 2

(c) EDFA 3 (d) EDFA 4

Fig. 6: PCA Failure detection performance over time.



(a) EDFA 1 (b) EDFA 2

(c) EDFA 3 (d) EDFA 4

Fig. 7: NLPCA Failure detection performance over time.

line in Fig. 4. The optimizer Adam was used for training, with the learning rate equal to 0.001 in a machine with 12.7GB RAM and an Intel(R) Xeon(R) CPU at 2.20GHz. The model's input, output, mapping, and de-mapping layers have 4 nodes each, while the bottleneck has only 1. Finally, the activation function for mapping, bottleneck, and de-mapping layers is ReLu, while it's Sigmoid for the output layer.

Regarding the training of the models, their entry is formed by the input power of the four EDFAs for each sample, $p = 4$ dimensions. For the reduction, after the fine-tuning, the number of components established is $k = 1$, since it guarantees the retention of at least 99.9% of data variance at reconstruction for both models, as presented in Fig. 5. Finally, in terms of threshold calculation, the chosen percentile is 99%, since it maintains a margin of 1%. That excludes large MSEs from non-failure samples with abnormal behaviors.

B. Comparison Results

The proposed approach is evaluated using the statistical concepts of Type I and Type II errors. In this context, a

TABLE I: Percentage of errors Type I and Type II for each EDFA by method.

	NLPCA		PCA	
	Type I(%)	Type II(%)	Type I(%)	Type II(%)
EDFA1	0.42	3.4	3.4	3.26
EDFA2	0.29	-	2.42	-
EDFA3	26.51	-	7.61	-
EDFA4	0.46	-	22.46	-
Average	6.92	3.4	8.97	3.26

Type I error occurs when normal condition data is incorrectly classified as a failure (false-positive), while a Type II error occurs when failure condition data is incorrectly classified as normal (false-negative). Among these, Type II errors are considered the more severe for the context under evaluation. The results obtained are represented in Table I as the percentile of errors. The general overview of anomaly detection over time is presented in Fig. 6 and Fig. 7 for each amplifier, with the outliers marked yellow.

The results verify that the average test accuracy is 90.19% for PCA and 92.23% for NLPCA. Also, the linear version triggered 2.05% more false alarms than the other on the tests, not being as accurate as the latter. However, PCA had only 0.14% fewer underreports of existing failures, a value that lacks statistical significance and is only marginally better.

Recognizing that different models may be biased towards minimizing one type of error, the possible impacts on the usage of each particular PCA-based algorithm are discussed. In an environment where Type II errors need to be reduced the most—given the extreme consequence of false-negative errors for optical networks—the best choice would be PCA. Otherwise, in specific situations where Type I is demanded to be low, NLPCA might be the best option.

Nonetheless, there might be situations where the goal is good cost-benefit performance. In that case, despite NLPCA taking longer to train with higher complexity than PCA, it is understood that PCA is the most efficient. Thus, although the non-linear version presented slightly better accuracy results, linear PCA is less computationally costly and can generate acceptable predictions.

Additionally, analyzing Table I, the plots in Fig. 6 and Fig. 7, it is evident that EDFA 3 was largely responsible for the number of failures in both models due to the data process acquisition for this dataset concerning its distribution for this EDFA. As seen in Fig. 3, for EDFA 3, the normal data during the training phase differed significantly from test data, even with both being made of data under normal conditions. Thus, it can be stated that it was not an issue caused by the model. However, EDFA 1 is the only one that exhibited failures, which leads to the interpretation, based on Table I, that EDFAs 2 and 3 were well-predicted with PCA, whereas EDFAs 2 and 4 had a better fit in the case of NLPCA.

V. Conclusions

This work introduced a comparative study between PCA techniques, revealing interesting nuances regarding their effectiveness in the simulated optical network scenario. Regarding the NLPCA results, a slight advantage over linear PCA was exhibited (2.05% in detection accuracy). However, this advantage was minimized due to the limited learning environment, resulting in NLPCA performance below expectations in EDFA3, substantially increasing false alarms. Additionally, linear PCA draws a lower number of Type II errors. As aforementioned, reducing this type of error is crucial for the optical link context. Nonetheless, considering the complexity of implementing each method and the results obtained for the specific scenario evaluated, NLPCA may be chosen as the more adequate model. These findings underscore the complexity of selecting the most suitable technique to optimize performance in specific application scenarios.

Acknowledgments

This work was supported in part by the National Council for Scientific and Technological Development (CNPq)/Eletronorte and by Amazon Sustainability Institute with Science and Innovation (ISACI).

Andrea Sgambelluri and Luca Valcarenghi from Scuola Superiore Sant'Anna, Italy, provided the dataset utilized in this study.

References

[1] Furdek, Marija, et al, Optical Network Security Management: Requirements, Architecture, and Efficient Machine Learning Models for Detection of Evolving Threats. Journal of Optical Communications and Networking 13.2: A144-A155, 2021.

[2] Y. Pointurier, Design of Low-margin Optical Networks. J. Opt. Commun.402 Netw., vol. 9, no. 1, pp. A9–A17, 2017.

[3] F. Musumeci et al, An Overview on Application of Machine Learning Techniques in Optical Networks. IEEE Communications Surveys and Tutorials, vol. 21, no. 2, pp. 1383-1408, Secondquarter, doi: 10.1109/COMST.2018.2880039, 2019.

[4] Shariati B, et al. Learning from the Optical Spectrum: Failure Detection and Identification. Journal of Lightwave Technology, 37: 433-440, 2019.

[5] X. Chen, B. Li, R. Proietti, Z. Zhu and S. J. B. Yoo, Self-Taught Anomaly Detection With Hybrid Unsupervised/Supervised Machine Learning. Optical Networks in Journal of Lightwave Technology, vol. 37, no. 7, pp. 1742-1749, doi: 10.1109/JLT.2019.2902487, 2021.

[6] Lun, Huazhi, et al. GAN-Based Soft Failure Detection and Identification for Long-haul Coherent Transmission Systems. 2021 Optical Fiber Communications Conference and Exhibition (OFC). IEEE, 2021.

[7] Wold, S., Esbensen, K., & Geladi, P., Principal Component Analysis. Chemometrics and intelligent laboratory systems, 2(1-3), 37-52, 1987.

[8] Hinton, G. E., & Salakhutdinov, R. R., Reducing the dimensionality of data with neural networks. Science, 313(5786), 504-507, 2006.

[9] Kramer, Mark A. Nonlinear principal component analysis using autoassociative neural networks. AIChE journal, v. 37, n. 2, p. 233-243, 1991.

[10] InRete Lab, Optical Failure Dataset. Scuola Superiore Sant'Anna, 2021. [Online]. Available: https://github.com/Network-And-Services/optical-failure-dataset