A Comparative Study on Semi-Supervised Learning Algorithms for Fault Management in Optical Networks

Ingrid Ramos, Andrei Ribeiro, Fabrício Lobato, Moisés F. Silva, and João C. W. A. Costa

***Abstract*—In recent years, the demand for reliable optical networks has intensified due to the rise of bandwidth-intensive applications. Faults in these networks can degrade transmission quality, leading to packet loss and service disruptions, making effective failure management essential for maintaining network availability and compliance with Service Level Agreements (SLAs). While most contemporary approaches rely on supervised learning (SL), which requires large amounts of often scarce fault data for training, this study explores semi-supervised learning (Semi-SL) algorithms for fault management in optical networks, which rely only on normal operating condition data, thus reducing the dependency on scarce fault data. We evaluate several cluster-based algorithms and dimensionality reduction techniques using a dataset from an optical testbed. Results based on FN/FP errors show that all techniques performed with an average accuracy exceeding 90%. The Autoencoder yielded the best fault management performance (96.46% of average accuracy), followed by Gaussian Mixture Model (92.5%), Mahalanobis Squared-Distance (92.48%), Density-based spatial clustering of applications with noise (92.42%), Fuzzy C-means (92.12%), K-means (92.09%), and Principal Component Analysis (90.22%).**

***Index Terms*—Fault management, Optical networks, Semi-supervised Learning, Cluster-based algorithms, Dimensionality reduction techniques.**

I. Introduction

OVER recent years, the rapid advancement of communication technologies, including 5G, the Internet of Things (IoT), and cloud computing, alongside the rise of new network services like VR/AR, autonomous vehicles, and 4K video, has led to an exponential increase in the demand for high-speed data transmission in optical communication networks [1]. Improving intelligence and versatility in optical networks is imperative to conquer this data rate as their complexity grows and new data-hungry applications emerge. Furthermore, the interconnected nature of modern systems means that any faults or disruptions in optical links can severely impact the quality of transmission (QoT), leading to data loss and breaches of service-level agreements (SLAs) [2]. Hence, promoting accurate and real-time fault management is essential to maintaining the reliability of optical network transmissions. In the same sense, the fulfillment of SLAs can be efficiently provided by ensuring effective monitoring of the physical layer regarding the occurrence of faults or degradation in the network [3].

Faults in optical networks are typically categorized into two types: soft failures and hard failures. These can arise from various causes, including component malfunctions, aging [4], physical layer attacks [5], and insufficient maintenance [6]. Hard failures, such as those caused by fiber cuts, result in immediate connection loss and significantly disrupt network transmissions. In contrast, soft failures are anomalies that cause notable variations in network performance but do not trigger explicit service disruption alarms [3]. Soft failures can occur in several optical devices, such as Transponders (TRX), Wavelength Selective Switches (WSS), and Optical Amplifiers (OA) [7]. If not addressed promptly, these failures can escalate into hard failures, jeopardizing multiple optical connections and the services they support. Therefore, detecting and pinpointing faults is vital for efficient network maintenance and cost reduction [8], [9].

Traditional fault management approaches rely on simplified threshold-based methods in the realm of ML-based fault management and probabilistic models [10], [11]. However, these methods are often inefficient, prone to errors, and labor-intensive, requiring skilled technicians to manage complex and dynamic scenarios. As optical networks expand in scale and complexity, implementing effective threshold-based strategies becomes increasingly challenging [12], [13]. The ultimate objective is to develop large-scale, fully automated networks capable of performing self-diagnostics based on policies tailored to the specific applications they support [14].

To address these challenges, machine learning (ML) and artificial intelligence (AI) techniques have gained significant attention for automating optical network management and control. Unlike traditional threshold-based methods, which often suffer from low fault detection rates and high false alarm rates [11], ML-based approaches present a promising solution. These methods are inherently data-driven, as their performance depends on the quality and quantity of data available for training [14], [15]. Data-driven strategies also enable rapid adaptation to evolving network conditions. By integrating these solutions, network operators can achieve autonomous, knowledge-based service provisioning [4], optimize resource allocation, ensure end-to-end QoT, and manage faults effectively [16]–[20].

Recent research has predominantly focused on supervised learning (SL) algorithms, which require labeled datasets with predefined correct outputs. In optical networks, however, datasets are often imbalanced, with significantly more data on

Ingrid Ramos, Andrei Ribeiro, Fabrício Lobato, and João C. W. A. Costa are with Institute of Technology, Federal University of Pará, Belém–PA, Brazil (e-mail: ingrid.ramos@itec.ufpa.br; andrei.ribeiro@itec.ufpa.br; frl@ufpa.br; jweyl@ufpa.br). ORCID: I. Ramos, 0009-0000-0393-2733; A. Ribeiro, 0009-0001-7452-8380; F. Lobato, 0000-0002-4467-7115; J. C. W. A. Costa, 0000-0003-4482-6886.

Moisés F. Silva is with Los Alamos National Laboratory, Los Alamos, NM, USA (e-mail: mfelipe@lanl.gov). ORCID: M. F. Silva, 0000-0001-7897-3978.

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).

Submission: 2025-08-20, First decision: 2025-10-17, Acceptance: 2025-12-30, Publication: 2026-01-09.

Digital Object Identifier 10.14209/jcis.2026.1.

normal conditions than on faults, which complicates model development and increases costs. While statistical models can create synthetic fault data, their practical implementation is challenging. A complementary direction leverages *digital twins* to synthesize labeled fault scenarios in silico, enabling fully supervised training without live fault injection in production networks [21]. However, semi-supervised learning (Semi-SL) techniques have emerged as a promising alternative for fault management in optical networks, as these models learn patterns from normal conditions and identify anomalies as deviations from these patterns [22], [23]. Even though labeled data is required, this simplifies data collection, as only normal-condition data is needed for training, and accelerates the deployment of fault management systems.

This work is an extension of a previously published paper that compared only two machine learning techniques: Principal Component Analysis (PCA) and its nonlinear version, Nonlinear Principal Component Analysis (NLPCA) [24]. In this subsequent analysis, we evaluate a broader range of algorithms, including Mahalanobis Squared Distance (MSD), Gaussian Mixture Model (GMM), Density-Based Spatial Clustering of Applications with Noise (DBSCAN), PCA, and Autoencoders (AEs), with the focus on clustering and dimensionality reduction techniques for fault detection and device-level localization.

The remainder of this paper is organized as follows. Section II relates this study to previous works. Section III provides the theoretical foundations of the seven compared algorithms. Section IV explains the operating principles of each algorithm and its application for fault detection and device-level localization in optical networks. Section V describes the optical testbed used for data acquisition, the dataset employed in our experiments, the performance results, and comparative analyses. Finally, Section VI concludes the paper.

II. Related Work

Vela et al. [22] proposed two finite state machine algorithms to detect causes of failures leading to significant bit error rate (BER) degradations in optical connections. The first algorithm analyzes pre-FEC BER data for gradual changes, while the second uses a probabilistic approach to identify the most likely failure among a range of classes. The study also examined machine learning-aided algorithms for soft failure localization in Elastic Optical Networks (EONs) [9], focusing on signal spectrum analysis to localize failures and estimate their severity.

Supervised learning methods, like SVM [25] and neural networks [26], rely on limited labeled fault data. In contrast, unsupervised learning techniques, such as Generative Adversarial Networks (GANs) [3] and autoencoders [7], address this challenge but often fail to localize faults. This study aims to fill that gap by comparing semi-supervised algorithms for fault detection and localization. Previous works primarily use Bit Error Rate (BER) characteristics for fault management in optical networks. At the same time, some, like Rafique et al. [26], apply neural networks to optical power measurements under various failure modes. This method identifies potential failures using block-based deviation before a neural network predicts abnormal behavior. Additionally, Mayer et al. [27] proposed a supervised neural network approach for localizing soft failures, relying on telemetry data from software-defined networks. However, it requires a numerical model of the network for simulating training data from failure scenarios.

Velasco et al. [28] proposed a soft-failure identification approach using an optical spectrum analyzer (OSA). They utilized a decision tree multi-class classifier to detect filtering failures and an SVM binary classifier to identify whether the failure resulted from Filter Shift or Tight Filtering. Additionally, three feature extraction methods were explored to improve failure detection related to filter cascading. In the same sense, Shimizu et al. [23] introduce a neural model that detects failures in optical links based on optical signal-to-noise ratio (OSNR) values from multiple lightpaths, providing a true/false response. However, this method only indicates the presence of a failure and does not address data imbalance or failure localization.

Most of the cited works rely on supervised learning, requiring data from both failure and normal conditions. However, in real-world optical networks, collecting failure data is often unfeasible and costly due to conservative design practices. As a result, failure data is scarce, while normal data is plentiful. This has led to a focus on unsupervised learning (UL) algorithms that utilize normal data effectively, even with imbalanced datasets. For instance, Lun et al. [29] introduced an unsupervised generative adversarial network (GAN) approach for soft-failure detection, using only normal samples for detection and supervised algorithms with failure samples for identification.

Similarly, authors in [30] proposed a hybrid unsupervised and supervised fault detection framework that combines density-based clustering techniques with deep neural networks. First, the approach uses the clustering technique to learn hidden patterns from unlabeled data and classify them into clusters corresponding to fault and normal conditions. Finally, the deep neural network is trained on labeled data generated by the clustering technique to perform anomaly classification. However, the proposed approach needs failure samples for the model training. It can also not perform failure localization, which is fundamental in rapidly repairing eventual failures in optical networks.

Kruse et al. [2] developed a VAE-GAN model for fault management, using Euclidean distance to differentiate between normal and failure data from optical spectrum analyzers. This framework effectively operates on limited training data and identifies unknown failure types, but it cannot perform fault localization since it focuses solely on failure detection and identification. Similarly, Liu et al. [31] proposed an autoencoder-based method for failure detection that requires only normal data for training, needing less than 3% of failure data for validation.

Silva et al. [32] presented a deep learning approach for multivariate time-series modeling that addresses fault detection, localization, and forecasting using a complex LSTNet architecture with convolutional and attention layers. While effective, this method is computationally intensive. In contrast,

Ribeiro et al. [33] tackle fault detection scalability with a cluster-based algorithm and PCA, but this approach lacks fault localization capabilities.

Several PCA-based approaches address data security concerns by leveraging their rotation-invariant property. For example, Silva et al. [34] use PCA for privacy-preserving soft-failure detection, identifying abnormal behaviors in telemetry data. Similarly, Sales et al. [35] introduced a disaggregated PCA method that enhances data confidentiality by distributing the learning across local models. However, while these methods focus on data security, they neglect fault localization issues.

Similarly, Shahkarami et al. [25] compared several machine learning algorithms for detecting and identifying equipment failures using BER data. They evaluated Binary Support Vector Machine (SVM), Random Forest (RF), Multiclass SVM, and a single hidden-layer neural network (NN) for failure detection based on complexity and accuracy, while a two-hidden-layer NN was used for failure identification. The results were promising, but the approach requires data from both failure and no-failure conditions for supervised learning.

III. Theoretical Background

A. Clustering Algorithms

1) K-means: K-means is a widely used algorithm for clustering large datasets. It is the simplest clustering technique, but it can be applied to a large range of clustering problems. A plethora of applications using K-means can be found in the literature, ranging from face detection to text processing [36], [37]. It works by assigning each data sample to one of the K clusters generated by the method. Also, it is considered a hard-clustering algorithm, meaning each sample belongs to only one cluster with the smallest Euclidean distance from the respective sample.

The main goal of K-means is to update the centroids by calculating the mean of the samples belonging to the cluster and repeating the relocation and updating process until the convergence criteria are satisfied. The user defines the number of clusters, and the initial positions of the clusters are generally random [38]. The procedures are shown in Algorithm 1.

K-means aims to minimize the objective function shown in line 7 of Algorithm 1, where $X = \{x_i\}_{i=1}^{N}$ is a dataset of N samples with $x_i \in \mathbb{R}^m$, and $\mu_k \in \mathbb{R}^m$ is the centroid of cluster k. The algorithm iteratively updates the centroids over T iterations to converge to a local minimum. The indicator variable $r_{ik} \in \{0, 1\}$ specifies whether sample x_i is assigned to cluster k.

The number of clusters defined by the user is a key factor that directly affects the performance of K-means. This dependence presents a limitation of the model, as it often relies on the user to specify the number of clusters in advance, necessitating expert domain knowledge about the underlying datasets. This requirement becomes especially problematic with large-scale datasets, where the complexity and variability can significantly impact the performance of the algorithm. In response to this limitation, several statistical criteria (e.g., Akaike Information Criterion (AIC)) have been investigated, providing an automatic selection of the appropriate number of clusters [39]. Therefore, AIC was employed in this study.

Algorithm 1 K-means Clustering Algorithm

1: **Input:** Data points $\{x_i\}_{i=1}^{N}$, number of clusters K, number of iterations T
2: **Initialize** cluster centers $\{\mu_k^0\}_{k=1}^{K}$ (e.g., randomly)
3: **for** each iteration $t = 1$ to T **do**
4: **Assignment step:**
5: **for** each data point x_i **do**
6: **for** each cluster $k = 1$ to K **do**
7: **if** $k == \arg\min_j \|x_i - \mu_j^{(t-1)}\|$ **then**
8: $r_{ik} = 1$
9: **else**
10: $r_{ik} = 0$
11: **end if**
12: **end for**
13: **end for**
14: **Update step:**
15: **for** each cluster $k = 1$ to K **do**
16: $\mu_k^t = \frac{\sum_{i=1}^{N} r_{ik} x_i}{\sum_{i=1}^{N} r_{ik}}$
17: **end for**
18: **end for**
19: **Output:** Final cluster centers $\{\mu_k^T\}$ and assignments $\{r_{ik}\}$

2) Fuzzy C-means: FCM is a centroid-based clustering algorithm similar to K-means [40]. Conversely, FCM belongs to the soft-clustering (or fuzzy clustering) category. Unlike hard-clustering, which presents limitations when a single data point has patterns belonging to dissimilar clusters, soft-clustering algorithms reduce these limitations by providing more information about the memberships of the data points [41]. In the FCM method, the sample may belong to all clusters with a certain degree of fuzzy membership [42].

The advantage of FCM is the generation of new clusters from the data points with close membership values to existing classes [43]. In that regard, the equation for the centroids of each cluster refers to a weighted average, given by:

$$\mu_k = \frac{\sum_{i=1}^{N} w_{ik}^p x_i}{\sum_{i=1}^{N} w_{ik}^p} \tag{1}$$

where w_{ik} denotes the membership degree of sample $\boldsymbol{x}_i$ to cluster k, and p is the fuzziness parameter that controls the smoothness of the resulting partition. Membership values are initially set randomly within the range allowed by the algorithm (0 to 1) and updated after calculating the cluster center. The updating of the membership values matrix is given by Equation 2, as it follows:

$$w_{ik} = \left(\sum_{j=1}^{C} \left(\frac{\|x_i - \mu_k\|}{\|x_i - \mu_j\|} \right)^{\frac{2}{p-1}} \right)^{-1} \tag{2}$$

The FCM algorithm updates the membership values of C clusters, minimizing J in Equation 3 until it reaches a local minimum or a pre-defined number of iterations. Note that K-means can be seen as a particular case of FCM where membership values are binary, i.e., either 0 or 1.

$$J = \sum_{i=1}^{N} \sum_{K=1}^{C} w_{ik}^{p} \|x_i - \mu_k\|^2. \quad (3)$$

The steps of the FCM algorithm can be summarized in the following algorithm 2:

Algorithm 2 Fuzzy C-Means Clustering Algorithm

Input: Data points $\{x_i\}_{i=1}^{N}$, number of clusters C, fuzziness parameter p, number of iterations T
Initialize cluster centers $\{\mu_k^0\}_{k=1}^{C}$ (e.g., randomly)
Define a distance metric $d(x_i, \mu_k)$
for each iteration $t = 1$ to T **do**
    Update membership matrix:
    for each data point x_i **do**
        for each cluster $k = 1$ to C **do**
            Compute $d(x_i, \mu_k^{(t-1)})$
            Compute $w_{ik}^{(t)}$ using Equation 2
        end for
    end for
    Update cluster centers:
    for each cluster $k = 1$ to C **do**
        Update μ_k^t using Equation 1
    end for
end for
Output: Final cluster centers $\{\mu_k^T\}$ and membership matrix $W^{(T)}$

3) Mahalanobis squared-distance: MSD is a well-known correlation-based statistical distance metric that measures the dissimilarity between two multivariate data sets [44], [45]. Unlike the Euclidean distance, it considers the correlation between the parameters, not depending on the scale of the features. However, such a model assumes that data follows a unique multivariate Gaussian distribution; thereby, the data can be modeled by a single Gaussian cluster, as follows 4.

$$MSD(x_i) = (x_i - \mu)\Sigma^{-1}(x_i - \mu)^T \quad (4)$$

for X with mean μ and covariance matrix Σ^{-1}.

4) Gaussian Mixture Model: GMM is a model-based data clustering algorithm that overcomes the limitations of MSD. GMM allows the form of multiple ellipsoidal-shaped clusters or components based on probability density estimations using the Expectation-Maximization (EM) [46], [47]. Each cluster is modeled as a Gaussian distribution. Unlike K-means, which is based only on the mean, GMM works with the mean and the covariance, providing a better quantitative measure of fitness per number of clusters. Multivariate finite mixture models capture the main clusters. Hence, the GMM can learn non-linear relationships, assuming that a finite multivariate Gaussian distribution can model the data. For a GMM, each component $g(x_i|\theta_k)$ is represented as a Gaussian distribution,

$$g(x_i|\theta_k) = \frac{\exp -\frac{1}{2}(x_i - \mu_k)^T \Sigma_k^{-1}(x_i - \mu_k)}{(2\pi)^{m/2}\sqrt{\det(\Sigma_k)}}, \quad (5)$$

where each component is denoted by the parameters, $\theta_k = \{\mu_k, \Sigma_k\}$, the mean vector, μ_k and the covariance matrix, Σ_k. Thus, a GMM is completely specified by a set of parameters $\Theta = \{\alpha_1, \alpha_2, ..., \alpha_K, \theta_1, \theta_2, ..., \theta_K\}$. Hence, a finite mixture model $g(x_i|\Theta)$ is the weighted sum of $K > 1$ components $g(x_i|\theta_k)$ in $\mathbb{R}^m$,

$$g(x_i|\Theta) = \sum_{k=1}^{K} \alpha_k g(x_i|\theta_k), \quad (6)$$

where α_k corresponds to the weight of each component. These weights are positive $\alpha_k > 0$ with $\sum_{k=1}^{K} \alpha_k = 1$.

The EM local search method is one of the most used methods for estimating the GMM parameters [47], [48]. This method consists of two steps: i) expectation and ii) maximization. For the log-likelihood (LogL), $\log(g(X|\Theta)) = \log(\prod_{i=1}^{N} g(x_i|\Theta))$, the two steps are applied alternately and converge to a local optimum [47]. The choice of initial parameters Θ directly impacts the performance of the EM algorithm, as a wrong choice of the initial parameters can result in many replications of this method during execution.

5) Density-based spatial clustering of applications with noise: Unlike the previous algorithms, DBSCAN forms clusters based on the density of the data [49]. Datapoints with similar densities will belong to the same cluster.

One of the major benefits of DBSCAN is that this algorithm does not require presenting the number of clusters, as the model finds this parameter by itself. Instead of centroids, DBSCAN forms clusters by finding the core points. Two user-chosen parameters ϵ and MinPts, which mean the radius of the neighborhood and minimum number of points in the ϵ-neighborhood of a core point, respectively, are the most relevant parameters for DBSCAN. The minimum number of neighbors within a radius ϵ (with a user-chosen distance measure) is used to estimate the minimum density level. Samples that satisfy the minimum density conditions are considered a core point. All neighbors within the ϵ radius are considered part of the same cluster as the central point. Points that are not center points and outside the radius of a center point are considered noise and do not belong to any cluster.

According to the distribution information of the data, this algorithm identifies regions with different densities and generates the cluster results [49]. The DBSCAN algorithm steps are summarized as follows in Algorithm 3.

In Algorithm 3, the parameter ϵ defines the radius of the neighborhood around each point, while $MinPts$ is the minimum number of neighboring points required to consider a point as a core point. Points that do not meet this density condition and are not reachable from any core point are labeled as noise.

B. Dimensionality Reduction Algorithms

1) Principal component analysis: PCA is a widely used technique in ML and statistics for dimensionality reduction and data visualization [50]. 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 [51], [52]. This transformation is achieved by finding the eigenvectors and eigenvalues of the covariance matrix.

Algorithm 3 DBSCAN Clustering Algorithm

Input: Dataset $X = \{x_i\}_{i=1}^{N}$, neighborhood radius ϵ, minimum number of points $MinPts$
Initialize: All points in X are unvisited
for each data point $x_i \in X$ **do**
  if x_i is unvisited **then**
    Mark x_i as visited
    Retrieve the ϵ-neighborhood of x_i:
    $L = \{x_j \in X \mid d(x_i, x_j) \leq \epsilon\}$
    if $|L| \geq MinPts$ **then**
      Create a new cluster and add x_i and all points in L to it
      for each $x_j \in L$ **do**
        if x_j is unvisited **then**
          Mark x_j as visited
          Retrieve $L_j = \{x_k \in X \mid d(x_j, x_k) \leq \epsilon\}$
          if $|L_j| \geq MinPts$ **then**
            $L \leftarrow L \cup L_j$
          end if
        end if
        if x_j is not yet assigned to any cluster **then**
          Add x_j to the current cluster
        end if
      end for
    else
      Mark x_i as noise
    end if
  end if
end for
Output: Identified clusters and noise points

In general, given X with N observations and m 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, being the centered data matrix denoted as X'. Next, PCA computes the covariance 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 eigenvector decomposition is computed. The principal components are obtained by solving the eigenvalue equation:

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}, \tag{7}$$

where $\mathbf{A}$ is the covariance matrix, $\mathbf{v}$ is an eigenvector, and λ its associated eigenvalue. To find the eigenvectors, one must compute the eigenvalues λ by solving the determinant equation:

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0, \tag{8}$$

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 $\mathbf{v_1}$, $\mathbf{v_2}$, ... $\mathbf{v_m}$ and eigenvalues $\lambda_1, \lambda_2, ..., \lambda_m$ 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 d PCs corresponding to the d largest eigenvalues represent the most important directions of variation in the data.

Specifically, 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. Leveraging such ability, PCA compresses the dimensions of the data into a lower dimension with the minimum losses possible, thereby being useful for scenarios where the number of features tends to be large.

2) Autoencoder: AEs refer to neural network-based models trained to copy their input to their output [53]. These models can reduce the dimensionality of the data by encoding the input features into a compressed representation comprising the most significant relationships within the data distribution. The same model can reverse the compressed data to the original space without significant information loss. Unlike PCA, AEs use non-linear mapping functions that better learn implicit information from the data in a latent dimensional space [53].

In its most general form, AEs comprise an auto-associative neural network with three hidden layers, where the input features are the output targets, and the middle layer (also referred to as the bottleneck) is designed to learn the compressed representations of the input variables [54]. Although one may consider the ideal AEs to succeed in perfectly copying their inputs, it is not useful. In this case, the ultimate goal is to copy only the most valuable aspects of the input variables, constraining the number of hidden units to be less than the number of features, which allows the learning of useful properties of the data.

Diving into the encoder-decoder process of the model, as shown in Fig. 1, the hidden layer describes a code used to represent the input by performing a mapping $\boldsymbol{h} : \Re^m \rightarrow \Re^d$ of the input $\boldsymbol{x}$. After, the data are reversed to the original space by a demapping operation $\boldsymbol{g} : \Re^d \rightarrow \Re^m$. The learning process aims to find the set of parameters $\Theta = \{\theta, \theta'\}$ that minimizes the loss function:

$$L(\Theta) = \frac{1}{N} \sum_{\forall \boldsymbol{x} \in \boldsymbol{X}} \left\| \boldsymbol{x} - g_{\theta'}(h_\theta(\boldsymbol{x})) \right\|^2, \tag{9}$$

where $L(\Theta)$ is a loss function penalizing $g_{\theta'}(h_\theta(\boldsymbol{x}))$ for being dissimilar from $\boldsymbol{x}$, i.e, a mean square reconstruction error.

The most common approach for the encoder and decoder is through affine mappings and nonlinear functions:

$$\begin{aligned} h_\theta(\boldsymbol{x}) &= s_h(\theta\boldsymbol{x} + \boldsymbol{b}), \\ g_{\theta'}(\boldsymbol{x}) &= s_g(\theta^T\boldsymbol{x} + \boldsymbol{a}). \end{aligned} \tag{10}$$

Thus, the set of parameters turns out into $\Theta = \{\theta, \boldsymbol{b}, \boldsymbol{a}\}$, where $\boldsymbol{b}$ and $\boldsymbol{a}$ are the bias and θ is the weight matrix. This scheme of shared weights is called symmetric architecture [50].

One can note that by removing the mapping and demapping layers, an autoencoder (AE) reduces to a single hidden layer, resembling linear PCA when its nodes are linear. Moreover, the performance of an AE network with only one bottleneck layer of sigmoidal nodes is often no better than that of linear PCA, as using a sigmoidal function only allows for the representation of linear combinations of the inputs compressed by the sigmoid into the range (-1, 1) [54]. Additionally, while traditionally used for dimensionality reduction, AEs can also be adapted for tasks like classification, clustering, and anomaly detection, learning a subspace similar to principal components [50].

Parallel to the advances in deep learning models, additional autoencoder-based approaches have been made by combining simple autoencoders to create deeper models. These individual autoencoders can be independently trained for specific tasks, rather than training a whole network for a global task [54].

The selection of these seven algorithms is motivated by their strong representativeness and frequent adoption in optical network fault management studies. These methods encompass two key methodological categories—clustering and dimensionality reduction—enabling a comprehensive comparison within the main paradigms applied to semi-supervised fault detection.

IV. Proposed Approach

A. Semi-supervised-learning approach for fault detection and localization in optical networks

The steps for applying the proposed Semi-SL-based techniques for fault detection and localization are highlighted in this section. Fig. 1 summarizes the main steps. As all the techniques work in a semi-supervised manner, only data from normal conditions is used for training. The telemetry is collected and pre-processed to be further fed into each model for training. Once the model is properly trained, fault indicators (FIs) are calculated for each sample in the training data. FIs are a key parameter for fault detection and device-level localization, indicating whether a sample is derived from a fault or normal condition. Given a sample with m dimensions corresponding to the telemetry parameters, the respective FI for this sample is a vector with m dimensions, where each value indicates whether a telemetry parameter exhibits faulty behavior. Leveraging this, fault localization can be performed as an eventual fault can be detected individually for each telemetry parameter.

As shown in Fig. 1, the FIs calculation is different for each type of Semi-SL model. For the cluster-based models, except for the distribution-based models, the FIs are calculated through the generated centroid values. For PCA and AE, the FIs are the reconstruction errors (REs) generated from the input reconstructions at the output. The following topics highlight the procedures for FI calculation for each proposed method:

- ***FI for K-means, FCM, and DBSCAN.*** Considering a dataset $X_{n\times m}$, with n samples of m dimensions, the K-means FIs are given using the following Equation:

$$FI(x_i, \mu_j) = \sqrt{\sum_{p=1}^{m} (x_{ip} - \mu_{jp})^2}, \tag{11}$$

  where μ_j is the closest centroid to the sample.
  The FIs for the FCM are the same as those for K-means since both algorithms are based on the same clustering approach. For the FCM, FIs are given using the Equation 11. Similarly, the calculation of FIs for the DBSCAN leverages the same equation with only one difference. The only difference is that this one uses the final core points instead of centroids. DBSCAN calculates the FIs with the following equation:

$$FI(x_i, x_j) = \sqrt{\sum_{p=1}^{m} (x_{ip} - c_{jp})^2}, \tag{12}$$

  where c_j refers to the closest core point to the dataset sample x_i. As the models are trained using only data from normal conditions, FIs calculated from the training set have small values, because the centroids or core points were generated based on this data. Conversely, fault samples will generate large FIs as these are expected to present larger distances from the centroids or core points.
- ***FI for MSD and GMM.*** The FIs for MSD and GMM differ from those of other algorithms presented. In this case, the Mahalanobis squared distance calculates the FI. Unlike the Euclidean distance, it takes into account the mean μ_k and covariance matrix Σ_k^{-1} related to the distribution of the data, for which the FIs are given using the following equation:

$$FI(x_i|\theta_k) = (x_i - \mu_k)\Sigma_k^{-1}(x_i - \mu_k)^T, \tag{13}$$

  where θ_k is the closest component to the sample.
- ***FI for PCA and AE.*** Either PCA or AE can be used for anomaly detection tasks. In that case, the model leverages the reconstruction error (RE), which can be calculated using the Mean Squared Error (MSE) equation. Once the model is trained with only data from normal conditions, REs for such conditions will present small values compared to those computed from data from fault conditions. Hence, REs act like FIs for the fault detection task. In practice, REs can be extracted from the loss function described in Eq. 9. Given an input datapoint $\boldsymbol{x}$ and a reconstructed input datapoint $\hat{\boldsymbol{x}}$ predicted at the output, the FI can be computed following the formula 15:

$$FI = \sum_{\forall \boldsymbol{x} \in \boldsymbol{X}} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 . \tag{14}$$

Following the next procedures in Fig. 1, once all the training-derived FIs are computed, linear thresholds are calculated for each telemetry parameter that composes the dataset.

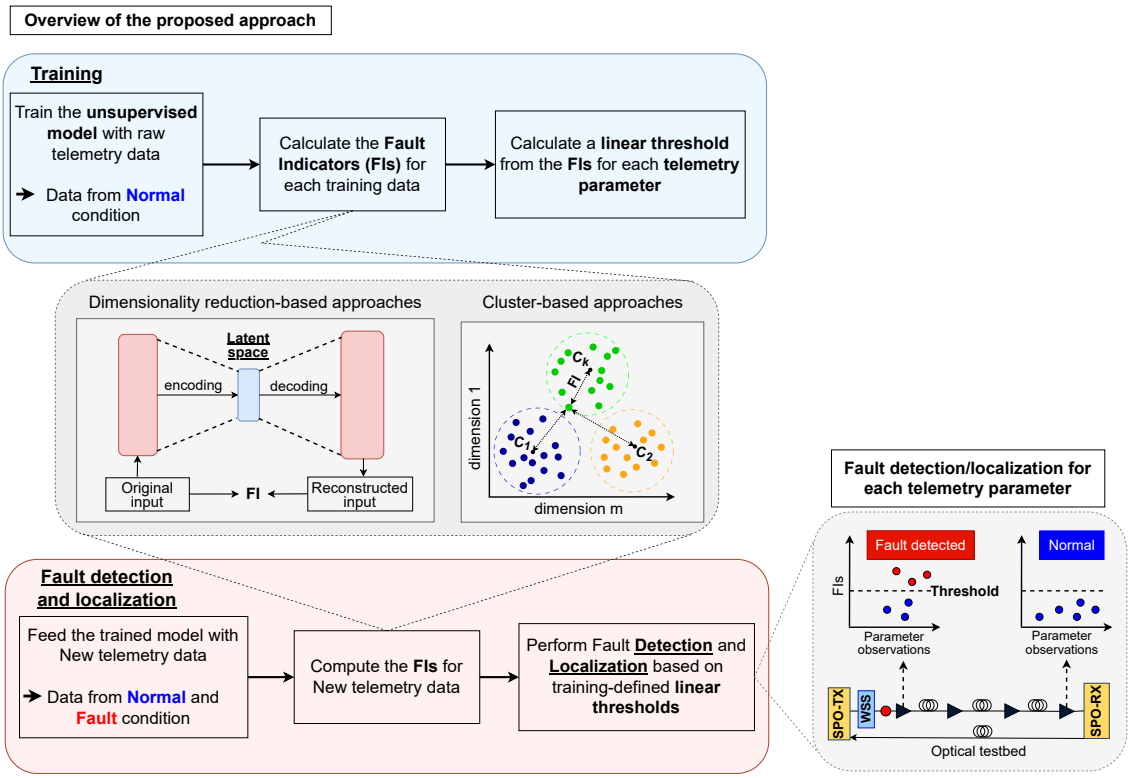


Fig. 1: overview of the proposed Semi-SL-based approach for fault detection and localization.

These thresholds are calculated based on a pre-defined percentile value from the FIs. Data from normal and fault conditions is used for the test phase. The proposed UL models aim to accurately distinguish between normal and fault samples. Hence, the trained model computes new FIs for every sample in the testing dataset.

In the next step, the FIs derived from the testing dataset are evaluated under the training-defined thresholds. Note that each FI is a vector comprising values corresponding to each telemetry parameter, as each FI is computed based on telemetry data collected from the optical network testbed. Thus, each feature value is analyzed for its respective threshold defined in the training phase. Hence, as shown in Fig. 2, if a sample of a specific telemetry parameter eventually presents an FI value higher than the threshold, the sample is classified as a fault condition, indicating that the fault is located at the respective telemetry parameter; otherwise, it is normal.

B. Model fine-tuning

The correct mapping of the network's normal condition is crucial to ensure effective fault management. For this reason, it is crucial to pass all the models through a fine-tuning process. In this case, the process aims to find the optimal hyperparameter values that yield the best results in terms of accuracy and model computational complexity.

Different UL-based models use different methods to find their optimal hyperparameter values. For cluster-based models, an appropriate number of clusters is necessary to enable the models to better capture the normal behavior of the network. For MSD, no optimization is required as only one cluster or component can be generated in that approach. For DBSCAN, several training trials are conducted to evaluate different values of ϵ and $MinPts$. The same procedure was applied to PCA, varying the number of principal components. The best values are those that achieved the best fault detection performance during the training phase, where only data from normal conditions is used.

For the remaining models, a typical fine-tuning method can be employed: the Akaike information criterion (AIC) [39], [50]. AIC refers to a function expressing the trade-off between fitting accuracy and the number of adjustable parameters. For K-means, FCM, and GMM, the AIC uses the negative log-likelihood and adds a penalizing term associated with the number of parameters, penalizing more complex models. The AIC is given by:

$$AIC = -2L(A) + 2v(A), \tag{15}$$

where $L(A)$ is the log likelihood function and $v(A)$ is the number of free parameters in the clustering model A. Several trials with different numbers of clusters are conducted. The location of the minima in AIC defines the best model. The AIC equation is similar to the AE model, with a few changes. The AIC, in this case, is given by:

$$AIC = \ln(e) + \frac{2N_w}{X}, \tag{16}$$

where $N_w = (s + f + 1)(M_1 + M_2) + s + f$ is the number of weights, where s is the number of nodes (neurons) in

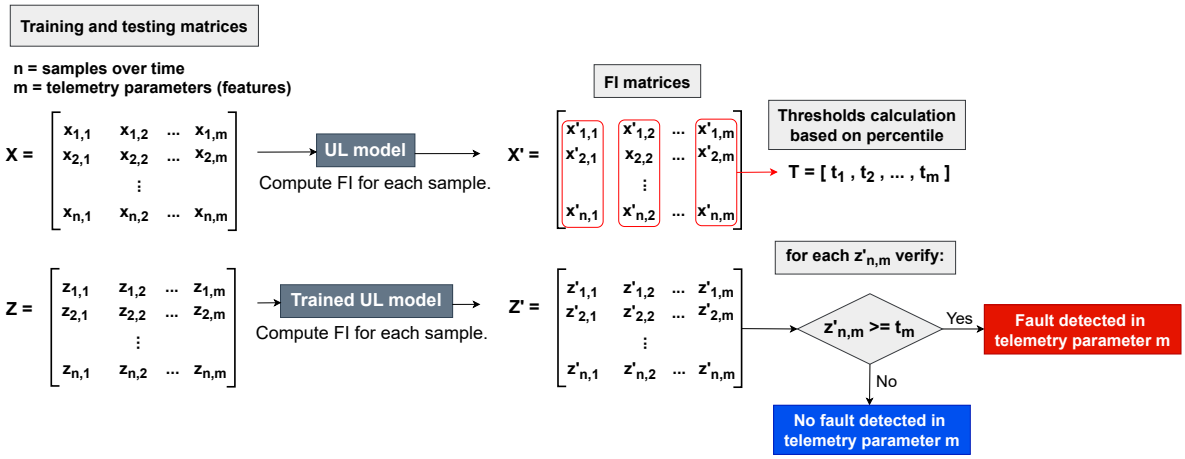


Fig. 2: Fault localization scheme of the proposed approach.

the input and output layer, f is the number of nodes in the bottleneck layer (latent space), M_1 and M_2 are the number of nodes in mapping (encoding) and demapping (decoding) layers, respectively. $X = Nxm$ is the number of entries in the data matrix, and $e = E/(2X)$ is the average sum of squares error. Minimization of these functions identifies AE models that are neither over- nor under-parameterized. In this paper, several trials with different AE architectures (i.e., varying the number of nodes in the encoder, latent space, and decoder layers) are conducted to determine the optimal AE node architecture.

V. Results and Discussion

This section presents the fault detection of the compared models. The optical testbed used to generate the dataset evaluated in this work is also detailed. Moreover, the procedures for data pre-processing are highlighted, including the selection of telemetry parameters, the splitting of the data into training and testing sets, and the fine-tuning of the compared models.

A. Experimental setup and data acquisition

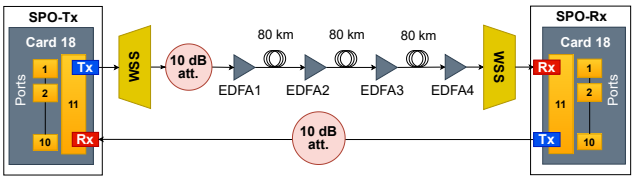


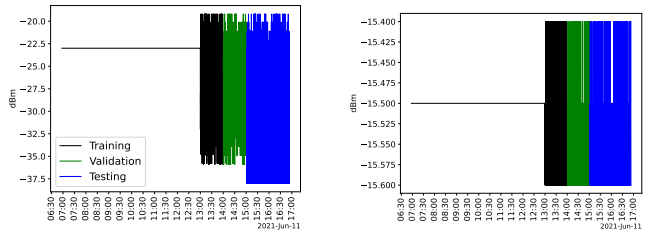
Fig. 3: Optical network testbed.

The ARNO testbed is from the Tecip Institute of Scuola Superiore Sant'Anna in Pisa, Italy, and can be found in [55]. Fig. 3 shows the optical devices that compose this testbed. It includes two Ericsson SPO 1400 devices, one Wavelength Selective Switch (WSS), and four EDFA amplifiers. These EDFAs are used to compensate for the power attenuation experienced along each traversed span. All the EDFA amplifiers are controlled via SPO devices (ampli1 and ampli2 are controlled by SPO-Tx, while SPO-Rx controls ampli3 and ampli4) and present a configurable gain in the range 15-25dB, with output mute power of 0.4 dBm. All the amplifiers are configured in constant gain mode, with a gain value that allows each span to be entered with 0dBm of optical power. Each SPO is equipped with a 100Gb/s Optical Transport Network (OTN) muxponder (installed at slot 18) with a DWDM optical line (port 11) and 10 tributary ports. The output of the first SPO (SPO-Tx) has been attached to a WSS, which is then attached to a multi-span link over a 10dB attenuator. The optical link between the SPO-Tx and SPO-Rx consists of 3 spans of 80 km each. The data is collected with a sampling frequency of 3.5 seconds over 10 hours. In the first 8 hours, two normal operation conditions 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 in the range from 0 to 18 dB. In the remaining 2 hours, the same behavior as during 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 correctly again. To collect the data, the monitoring system proposed in [56] has been used. By selecting the appropriate metric at each traversed element (i.e., the input/output optical power levels at each amplifier and the OSNR and the BER at the transponders), we relied on the Kafka-based telemetry system to distribute the selected metrics, while a new collector (i.e., Kafka consumer, receiving the telemetry) has been designed to export the data in a CSV format.

B. Dataset pre-processing and evaluation metrics

The dataset comprises samples collected at approximately 3.5-second intervals over a total of 10 hours, limited by the minimum time required by the commercial equipment to provide coherent transmission data (i.e., BER).

An exploratory data analysis identified significant missing values that could impact the performance of the proposed techniques. To address this, a spline linear interpolation technique [56] was applied, resulting in a final dataset of 13,948 samples. Only the input power data from the EDFA was used



(a) EDFA1 input power (b) EDFA2 input power

(c) EDFA3 input power (d) EDFA4 input power

Fig. 4: Optical dataset along the training, validation, and testing sets.

for localization, with 80% allocated for training and 20% for testing, as shown in Fig. 4. Notably, the EDFA1 input power data exhibited clear drops in magnitude, indicating expected outliers related to this equipment. Simulated failures occurred in the last two hours of testing, corresponding to 711 samples.

Soft failures were injected into the system by applying 25 dB of attenuation, which reduced the received signal below the receiver's sensitivity. The training and validation datasets only included normal conditions, enabling novelty detection to identify deviations as potential outliers. The data partitioning is illustrated in Fig. 4.

Fault detection performance was evaluated using FP (false-positive) and FN (false-negative) error rates, with threshold values based on the 99% percentile from the training data, as detailed in Section IV.

C. Fine-tuning of the compared models

A critical aspect of working with ML models is the fine-tuning process. From the experimental setup, we conducted intensive experiments using the previously described models to determine their optimal hyperparameters.

For DBSCAN, several values for its two main parameters were evaluated. The values for ϵ and $MinPts$ were: $[0.1, 0.001, 0.0001, 0.00001, 0.000001]$ and $[5, 50, 100, 800, 1500, 2000]$, respectively. Based on the training accuracy, the best values correspond to $\epsilon = 0.00001$ and $MinPts = 1500$. For the PCA model, the optimal number of principal components (PCs) is required as PCA reduces the dimensionality of the data using these parameters. There are several ways to find that value, but this work defines them based on the retaining variance. More specifically, different numbers of PCs ranging from 1 to 3 were tested to find which value could retain the maximum variance of the data when reduced dimensionality (Fig. 5). From these tests, one can note in Fig. 5 that when reducing the dimensionality of the data (initially having four features corresponding to the four EDFAs) to a 1-dimensional space (using PCs = 1), approximately 99% of the variance of the data is retained. Therefore, the final PCA model has only one dimension in the latent space.

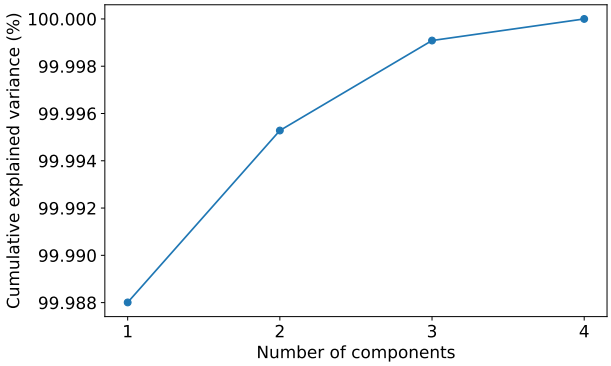


Fig. 5: Explained variance retained by PCA per number of components.

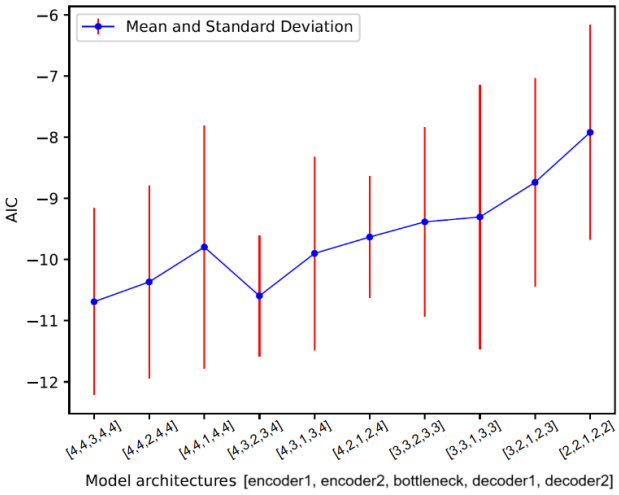


Fig. 6: Average AIC values of different AE node combinations.

Ten node combinations with different numbers of neurons for five layers (two encoders, one bottleneck, and two decoders) were tested on 20 training trials of 200 epochs each. At the end of each training, the mean squared error (MSE) is used in Equation 16 to calculate the AIC. The final AIC value for each combination of mapping and demapping nodes is the average AIC value from the 20 training trials. Fig. 6 presents the respective AIC values for each model architecture. Note that each architecture is represented in the figure by only the encoder and bottleneck layers, as the decoder layers are identical to the encoder, following a symmetric model architecture.

By analyzing the location of the minima in AIC presented in Fig. 6, the two models with the architectures [4,4,3] and [4,3,2] can be chosen as the most adequate models for our specific scenario. In that sense, the model with fewer neurons ([4,3,2]) was selected for further training over 20 trials, based on the minimum variation of training loss throughout a predefined maximum number of epochs. More specifically, the model

TABLE I: Fault detection results for all the compared models per EDFA device.

Device	EDFA1		EDFA2		EDFA3		EDFA4		Average
Model	FP (%)	FN (%)	FP (%)	FN (%)	FP (%)	FN (%)	FP (%)	FN (%)	
K-means	0.88	3.36	0	-	27.41	-	0	-	7.91
FCM	0.68	3.45	0	-	27.41	-	0	-	7.88
MSD	0.42	3.39	0	-	26.01	-	0.26	-	7.52
GMM	0	3.49	0	-	26.53	-	0	-	7.50
DBSCAN	0.42	3.39	0	-	26.01	-	0.26	-	7.58
PCA	3.39	3.28	2.41	-	7.60	-	22.45	-	9.78
AE	1.40	3.36	0.39	-	5.05	-	3.98	-	3.54

Note: "–" = N/A.

begins training and continues until the variation in training loss between actual epochs is smaller than a predefined value. It prevents the model from suffering *overfitting* by stopping the training when the model does not present a satisfactory decrease in the training loss.

In the cluster-based approaches, the AIC equation (Equation 15) was used to determine the optimal number of clusters for K-means, FCM, and GMM. Selecting the right number of components is crucial for proper model generalization. A range of clusters from 2 to 10 was evaluated over 20 trials for each model. Fig. 7 shows the average AIC results and their standard deviation.

As illustrated, K-means and FCM had nearly identical AIC values across components. Notably, all three models (K-means, FCM, and GMM) identified two components as optimal, which aligns with the two distinct data distributions in the training dataset. Thus, two clusters effectively represent the training data.

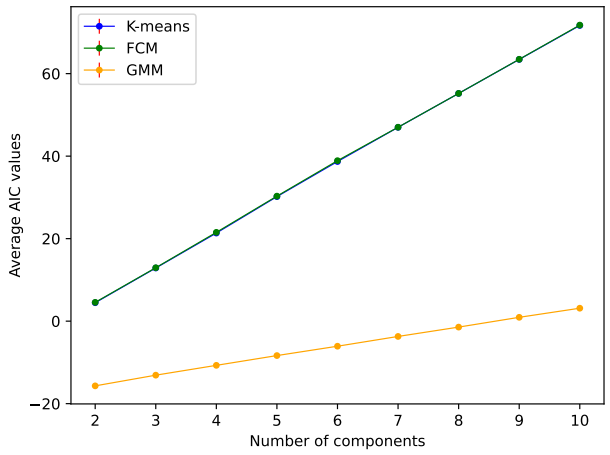


Fig. 7: Average AIC values of each cluster-based algorithm for various numbers of components.

D. Fault management results

For the evaluation of the presented approaches, Tab. I presents the FP and FN errors for each EDFA device in the testing set. FN errors are not shown for EDFA2, EDFA3, and EDFA4, as no failures were simulated for these devices. Overall, the cluster-based models exhibit marginal differences, with GMM showing slight superiority. For completeness, the accuracies on the testing set are: AE 96.46%, GMM 92.50%, MSD 92.48%, DBSCAN 92.42%, FCM 92.12%, K-means 92.09%, and PCA 90.22%. All models misclassified some failure samples in EDFA1, leading to FN errors. This misclassification is partly due to the failure simulation process, where samples labeled as "failure" often showed characteristics similar to normal conditions, making them difficult to identify.

Moreover, EDFA3 exhibits a high rate of FP errors, as illustrated in Fig. 4. Unlike other EDFAs, EDFA3's input power values show limited variation from normal conditions during the training period. This lack of data affects the clustering models' fault detection performance, reducing their generalization for test-phase fault scenarios.

For further evaluation, the FIs for the entire dataset from all cluster-based approaches are shown in Fig. 8. All models effectively detect faults in the EDFA1 equipment, achieving less than 3.50% of FN errors. The red dots in EDFA1 of each model represent faults, with most above the threshold line, corresponding to about 85% of the 711 fault samples. The dots below the threshold indicate the FN errors discussed earlier.

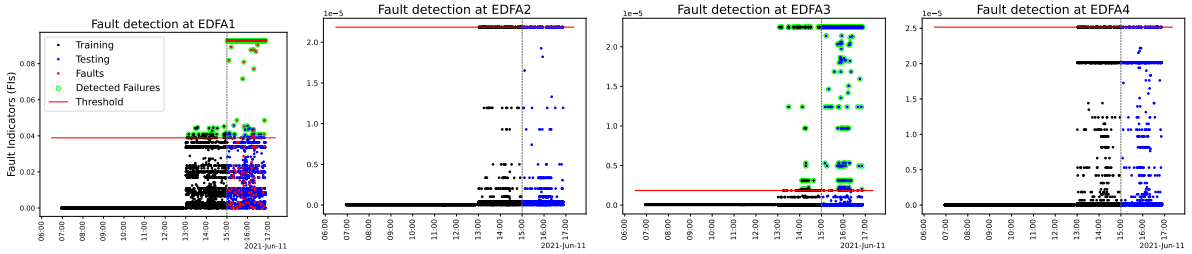
In fact, evaluations show marginal differences among cluster-based approaches, highlighting the need for a closer look at FP and FN errors. FP errors, where normal states are misclassified as faults, can increase operational expenses (OPEX). GMM is the most suitable option here due to slightly better performance. Conversely, FN errors indicate normality during faults, impacting network availability and SLAs. K-means may be preferable when availability is critical, as it performs better on FN errors.

For dimensionality reduction approaches, as seen in Tab. I, the PCA and AE models yield varied outcomes for each EDFA. For EDFA1, AE achieved a lower FP error rate (1.40%) compared to PCA (3.39%), while PCA obtained a slightly better FN error rate (3.28% versus 3.36%). For EDFA2, AE significantly outperformed PCA with a FP error of 0.39%, against 2.41% from PCA. A similar pattern is observed for EDFA3, where AE reached 5.05% FP error, markedly lower than PCA's 7.60%. These results suggest that AE better captures normal data variations across most devices, resulting in fewer false positives.

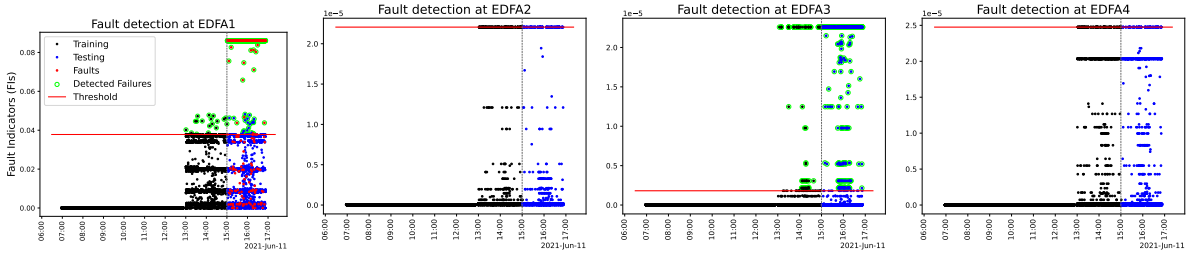
For EDFA4, AE significantly outperforms PCA, with PCA showing a FP error rate of 22.45%, misclassifying approximately 528 out of 2,353 samples from the testing set. In contrast, AE demonstrates only 3.98% FP errors. Fig. 8 illustrates the fault indicators (FIs) from both models across all EDFA parameters.

In EDFA1, as depicted in Fig.8, both models produced large

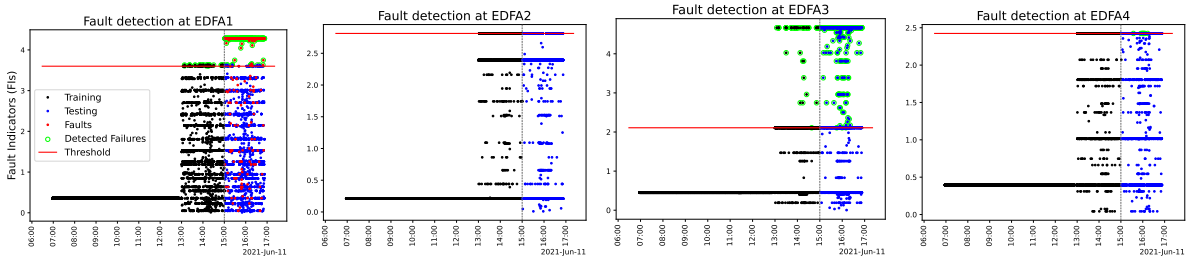
(a) K-means



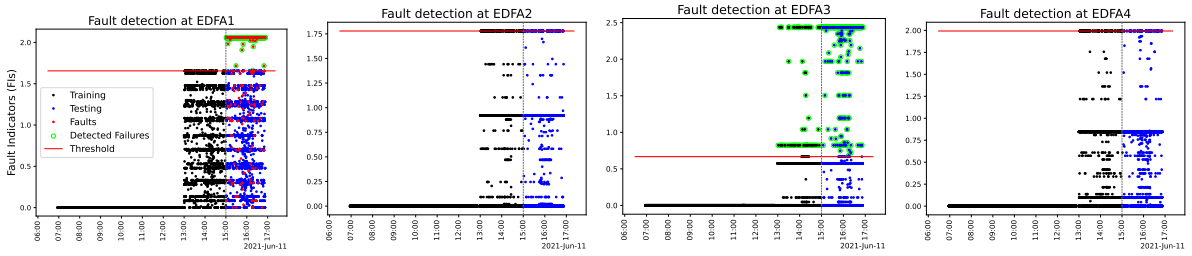
(b) FCM



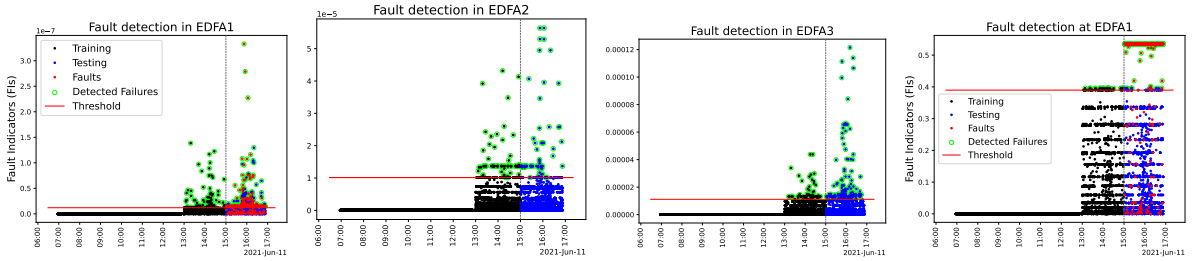
(c) MSD



(d) GMM



(e) PCA



(f) AE

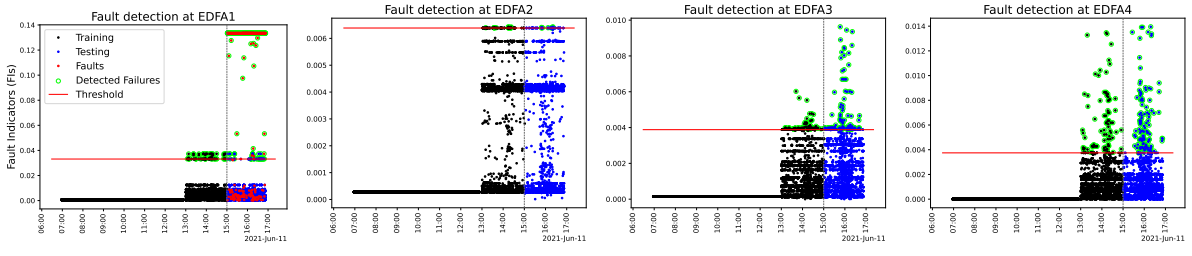


Fig. 8: Fault indicators along with a threshold defined over the training data for each cluster-based approach.

FIs for fault samples (red dots) compared to normal conditions (blue dots). PCA and AE effectively reconstructed normal condition inputs, resulting in small FIs, while significant reconstruction errors led to large FIs for fault samples. However, PCA's FIs for faults were only slightly different from those of normal samples, as evidenced by the overlapping red and blue dots.

For EDFA4, despite the visual similarity in FP errors (blue dots above the threshold) for both models, the actual numbers differ significantly, as detailed in Tab. I. If the threshold were raised, PCA's FP errors would decrease, with many blue dots falling below the threshold line.

In general, for the evaluated testbed, it can be verified that all the proposed semi-supervised learning-based approaches presented reasonable results regarding the minimization of misclassification. In this sense, the models could properly detect samples from fault conditions only by training with data from normal conditions. Among the cluster-based techniques, it is quite challenging to determine the best model, as Tab. I shows that the models yield very similar results. However, one can say that GMM performed marginally better, as the model correctly classified the samples under normal conditions in the testing set. The GMM modeling of the different distributions presented in the training data can explain these results.

Moreover, AE can be chosen as the most appropriate technique for the evaluated dataset, as this model presented stable and consistent results for fault localization, as well as the lowest average error percentage among all models, as shown in Tab. I at 3.54%. As AE works based on neural networks, it can potentially learn nonlinear distributions in the data, taking into account hidden relevant information from the data. Compared to PCA, the AE model can utilize nonlinear mapping functions to more accurately represent the data in the latent space, thereby better learning the variations presented in the data under normal conditions.

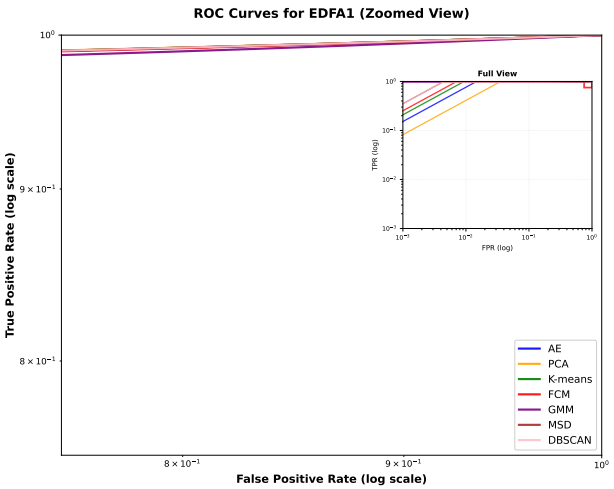


Fig. 9: ROC curves of the compared models.

Additionally, the ROC curves for EDFA1 are presented in Fig. 9 for all the compared models. On this device, a few clustering methods (MSD/DBSCAN and GMM) sit slightly closer to the upper-left corner, with lower FPR at a similar TPR, while the AE lands nearby with 1.40% FPR and 3.36% FNR. This difference is specific to EDFA1. Across all EDFAs, AE delivers the best average performance, with a much simpler, lighter architecture and minimal tuning. In practice, that consistency and low complexity make AE the most reliable choice overall, with the EDFA1 ROC providing an additional, device-level view where some alternatives happen to edge it out.

Furthermore, although neural network-based models are commonly more computationally expensive than cluster-based techniques (depending on the number of neurons and hidden layers), they do not represent a limitation for the particular fault detection task in optical networks. Additionally, the computational cost is irrelevant during the training phase, as the models are trained offline. Therefore, once trained, all the proposed techniques have similar computational costs in the testing phase, as the only procedure required in this phase is to calculate the FI for every sample from the testing dataset.

VI. Conclusion

This article has investigated the use of semi-supervised learning methods for fault detection and device-level localization using only normal operational data. Various clustering algorithms, including K-means, FCM, MSD, GMM, and DBSCAN, are compared with dimensionality reduction techniques such as PCA and AE. The findings reveal that while cluster-based algorithms effectively differentiate fault data from normal data, they exhibited inconsistencies in fault localization, resulting in a considerable number of false alarms. Conversely, AE outperformed other techniques in fault localization, achieving an average FP error rate of 2.7% and a FN error rate of 3.36%. This performance indicates AE's potential as the leading UL approach for optical monitoring systems where minimizing misclassifications is a priority. Based on these results, future work includes additional tests on larger datasets with different numbers of optical devices and telemetry parameters to assess scalability, and large language model (LLM)-based approaches integrated with retrieval augmented generation (RAG) techniques.

Acknowledgments

The authors thank A. Sgambelluri and L. Valcarenghi (Scuola Superiore Sant'Anna, Italy) for providing the dataset utilized in this study.

References

[1] R. Gu, Z. Yang, and Y. Ji, "Machine learning for intelligent optical networks: A comprehensive survey," *Journal of Network and Computer Applications*, vol. 157, p. 102576, 2020, doi:10.1016/j.jnca.2020.102576.

[2] L. E. Kruse, S. Kühl, A. Dochhan, and S. Pachnicke, "Experimental investigation of machine-learning-based soft-failure management using the optical spectrum," *Journal of Optical Communications and Networking*, vol. 16, no. 2, pp. 94–103, 2024, doi:10.1364/JOCN.500930.

[3] J. Rak, R. Girão-Silva, T. Gomes, G. Ellinas, B. Kantarci, and M. Tornatore, "Disaster resilience of optical networks: State of the art, challenges, and opportunities," *Optical Switching and Networking*, vol. 42, p. 100619, 2021, doi:10.1016/j.osn.2021.100619.

[4] X. Chen, R. Proietti, H. Lu, A. Castro, and S. B. Yoo, "Knowledge-based autonomous service provisioning in multi-domain elastic optical networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 152–158, 2018, doi:10.1109/MCOM.2018.1701191.

[5] C. Mas, I. Tomkos, and O. K. Tonguz, "Failure location algorithm for transparent optical networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 8, pp. 1508–1519, 2005, doi:10.1109/JSAC.2005.852182.

[6] R. Rejeb, M. S. Leeson, and R. J. Green, "Fault and attack management in all-optical networks," *IEEE Communications Magazine*, vol. 44, no. 11, pp. 79–86, 2006, doi:10.1109/MCOM.2006.248169.

[7] K. S. Mayer, J. A. Soares, R. P. Pinto, C. E. Rothenberg, D. S. Arantes, and D. A. Mello, "Machine-learning-based soft-failure localization with partial software-defined networking telemetry," *Journal of Optical Communications and Networking*, vol. 13, no. 10, pp. E122–E131, 2021, doi:10.1364/JOCN.424654.

[8] S. Barzegar, E. Virgillito, M. Ruiz, A. Ferrari, A. Napoli, V. Curri, and L. Velasco, "Soft-failure localization and device working parameters estimation in disaggregated scenarios," in *Optical Fiber Communication Conference*. Optica Publishing Group, 2020, pp. Th1F–2, doi:10.1364/OFC.2020.Th1F.2.

[9] A. Vela, B. Shariati, M. Ruiz, F. Cugini, A. Castro, H. Lu, and R. Proietti, "Soft failure localization during commissioning testing and lightpath operation," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 10, no. 1, pp. A27–A36, 2018, doi:10.1364/JOCN.10.000A27.

[10] D. Wang, C. Zhang, W. Chen, H. Yang, M. Zhang, and A. P. T. Lau, "A review of machine learning-based failure management in optical networks," *Science China Information Sciences*, vol. 65, no. 11, p. 211302, 2022, doi:10.1007/s11432-022-3557-9.

[11] F. Musumeci, "Machine learning for failure management in optical networks," in *Optical Fiber Communication Conference*. Optica Publishing Group, 2021, pp. Th4J–1, doi:10.1364/OFC.2021.Th4J.1.

[12] M. Zhang, C. You, H. Jiang, and Z. Zhu, "Dynamic and adaptive bandwidth defragmentation in spectrum-sliced elastic optical networks with time-varying traffic," *Journal of Lightwave Technology*, vol. 32, no. 5, pp. 1014–1023, 2014, doi:10.1109/JLT.2013.2296781.

[13] Y. Pointurier, "Design of low-margin optical networks," *Journal of Optical Communications and Networking*, vol. 9, no. 1, pp. A9–A17, 2017, doi:0.1364/JOCN.9.0000A9.

[14] G. Carrozzo, M. S. Siddiqui, A. Betzler, J. Bonnet, G. M. Perez, A. Ramos, and T. Subramanya, "Ai-driven zero-touch operations, security and trust in multi-operator 5g networks: A conceptual architecture," in *2020 European conference on networks and communications (EuCNC)*. IEEE, 2020, pp. 254–258, doi:10.1109/EuCNC48522.2020.9200928.

[15] M. Xie, J. S. Pujol-Roig, F. Michelinakis, T. Dreibholz, C. Guerrero, A. G. Sanchez, W. Y. Poe, Y. Wang, and A. M. Elmokashfi, "Ai-driven closed-loop service assurance with service exposures," in *2020 European Conference on Networks and Communications (EuCNC)*. IEEE, 2020, pp. 265–270, doi:10.1109/EuCNC48522.2020.9200943.

[16] R. Proietti, X. Chen, K. Zhang, G. Liu, M. Shamsabardeh, A. Castro, L. Velasco, Z. Zhu, and S. B. Yoo, "Experimental demonstration of machine-learning-aided qot estimation in multi-domain elastic optical networks with alien wavelengths," *Journal of Optical Communications and Networking*, vol. 11, no. 1, pp. A1–A10, 2019, doi:10.1364/JOCN.11.0000A1.

[17] S. Oda, M. Miyabe, S. Yoshida, T. Katagiri, Y. Aoki, T. Hoshida, J. C. Rasmussen, M. Birk, and K. Tse, "A learning living network with open roadms," *Journal of Lightwave Technology*, vol. 35, no. 8, pp. 1350–1356, 2017, doi:10.1109/JLT.2017.2660540.

[18] X. Chen, J. Guo, Z. Zhu, A. Castro, R. Proietti, H. Lu, M. Shamsabardeh, and S. B. Yoo, "Leveraging deep learning to achieve knowledge-based autonomous service provisioning in broker-based multi-domain sd-eons with proactive and intelligent predictions of multi-domain traffic," in *2017 European Conference on Optical Communication (ECOC)*. IEEE, 2017, pp. 1–3, doi:10.1109/ECOC.2017.8346218.

[19] B. Li, W. Lu, S. Liu, and Z. Zhu, "Deep-learning-assisted network orchestration for on-demand and cost-effective vnf service chaining in inter-dc elastic optical networks," *Journal of Optical Communications and Networking*, vol. 10, no. 10, pp. D29–D41, 2018, doi:10.1364/JOCN.10.000D29.

[20] J. Guo and Z. Zhu, "When deep learning meets inter-datacenter optical network management: Advantages and vulnerabilities," *Journal of Lightwave Technology*, vol. 36, no. 20, pp. 4761–4773, 2018, doi:10.1109/JLT.2018.2864676.

[21] S. Cruzes, "Optical network automation: Enhancing networks with machine learning, digital twins, and advanced technologies," TechRxiv preprint, September 2025, doi:0.36227/techrxiv.174140893.36394757/v2.

[22] A. Vela, M. Ruiz, F. Fresi, N. Sambo, F. Cugini, G. Meloni, L. Pot, L. Velasco, and P. Castoldi, "Ber degradation detection and failure identification in elastic optical networks," *IEEE/OSA Journal of Lightwave Technology*, vol. 35, no. 21, 2017, doi:10.1109/JLT.2017.2747223.

[23] D. Y. Shimizu, K. S. Mayer, J. A. Soares, and D. S. Arantes, "A deep neural network model for link failure identification in multi-path roadm based networks," in *2020 Photonics North (PN)*. IEEE, 2020, pp. 1–1, doi:0.1109/PN50013.2020.9166978.

[24] I. L. F. d. M. Ramos, A. C. Oliveira, A. Nogueira Ribeiro, R. Sales, F. Lobato, J. W. Costa, and M. F. Silva, "A comparative study on principal components analysis for failure detection in optical networks," *XLII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT2024)*, 2024, doi:10.14209/sbrt.2024.1571036264.

[25] S. Shahkarami, F. Musumeci, F. Cugini, and M. Tornatore, "Machine-learning-based soft-failure detection and identification in optical networks," in *2018 Optical Fiber Communications Conference and Exposition (OFC)*. IEEE, 2018, pp. 1–3, doi:10.1364/OFC.2018.M3A.5.

[26] D. Rafique, T. Szyrkowiec, H. Grießer, A. Autenrieth, and J.-P. Elbers, "Cognitive assurance architecture for optical network fault management," *Journal of Lightwave Technology*, vol. 36, no. 7, pp. 1443–1450, 2018, doi=10.1109/JLT.2017.2781540.

[27] K. S. Mayer, J. A. Soares, R. P. Pinto, C. E. Rothenberg, D. S. Arantes, and D. A. Mello, "Soft failure localization using machine learning with sdn-based network-wide telemetry," in *2020 European Conference on Optical Communications (ECOC)*. IEEE, 2020, pp. 1–4, doi:10.1109/ECOC48923.2020.9333313.

[28] L. Velasco, B. Shariati, A. P. Vela, J. Comellas, and M. Ruiz, "Learning from the optical spectrum: Soft-failure identification and localization," in *2018 Optical Fiber Communications Conference and Exposition (OFC)*. IEEE, 2018, pp. 1–3.

[29] H. Lun, X. Liu, M. Cai, Y. Wu, M. Fu, L. Yi, W. Hu, and Q. Zhuge, "Gan based soft failure detection and identification for long-haul coherent transmission systems," in *2021 Optical Fiber Communications Conference and Exhibition (OFC)*. IEEE, 2021, pp. 1–3, doi:10.1364/OFC.2021.Th4J.2.

[30] X. Chen, C.-Y. Liu, R. Proietti, J. Yin, Z. Li, and S. B. Yoo, "On cooperative fault management in multi-domain optical networks using hybrid learning," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 28, no. 4: Mach. Learn. in Photon. Commun. and Meas. Syst., pp. 1–9, 2022, doi:0.1109/JSTQE.2022.3151878.

[31] S. Liu, D. Wang, C. Zhang, L. Wang, and M. Zhang, "Semi-supervised anomaly detection with imbalanced data for failure detection in optical networks," in *Optical Fiber Communication Conference*. Optica Publishing Group, 2021, pp. Th1A–24, doi:10.1364/OFC.2021.Th1A.24.

[32] M. F. Silva, A. Pacini, A. Sgambelluri, and L. Valcarenghi, "Learning long-and short-term temporal patterns for ml-driven fault management in optical communication networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2195–2206, 2022, doi:10.1109/TNSM.2022.3146869.

[33] A. Ribeiro, R. Sales, F. Lobato, J. Costa, M. Silva, A. Sgambelluri, L. Valcarenghi, and L. Wosinska, "Pca-assisted fuzzy clustering approach for soft-failure detection in optical networks," in *2024 International Conference on Optical Network Design and Modeling (ONDM)*. IEEE, 2024, pp. 1–5, doi:10.23919/ONDM61578.2024.10582591.

[34] M. F. Silva, A. Sgambelluri, A. Pacini, F. Paolucci, A. Green, D. Mascarenas, and L. Valcarenghi, "Confidentiality-preserving machine learning algorithms for soft-failure detection in optical communication networks," *Journal of Optical Communications and Networking*, vol. 15, no. 8, pp. C212–C222, 2023, doi:10.1364/JOCN.481690.

[35] R. Sales, A. Ribeiro, M. Silva, F. Lobato, A. Sgambelluri, L. Valcarenghi, and J. Costa, "Disaggregated confidentiality-preserving scheme for fault detection in optical networks," in *2024 Optical Fiber Communications Conference and Exhibition (OFC)*. IEEE, 2024, pp. 1–3, doi:10.1364/OFC.2024.Th3I.8.

[36] M.-C. Su and C.-H. Chou, "A modified version of the k-means algorithm with a distance based on cluster symmetry," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 6, pp. 674–680, 2001, doi:10.1109/34.927466.

[37] M. Alhawarat and M. Hegazi, "Revisiting k-means and topic modeling, a comparison study to cluster arabic documents," *IEEE Access*, vol. 6, pp. 42 740–42 749, 2018, doi:10.1109/ACCESS.2018.2852648.

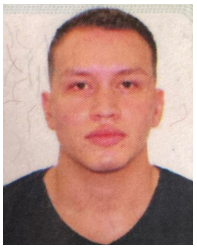
[38] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*, 1967.

[39] T. M. Kodinariya, P. R. Makwana *et al.*, "Review on determining number of cluster in k-means clustering," *International Journal*, vol. 1, no. 6, pp. 90–95, 2013.

[40] J. C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media, 2013, doi:10.1007/978-1-4757-0450-1.
[41] J. Nayak, B. Naik, and H. Behera, "Fuzzy c-means (fcm) clustering algorithm: a decade review from 2000 to 2014," in *Computational Intelligence in Data Mining-Volume 2: Proceedings of the International Conference on CIDM, 20-21 December 2014*. Springer, 2015, pp. 133–149, doi:10.1007/978-81-322-2208-8$_1$4.
[42] M. R. Ferreira and F. D. A. De Carvalho, "Kernel fuzzy c-means with automatic variable weighting," *Fuzzy Sets and Systems*, vol. 237, pp. 1–46, 2014, doi:10.1016/j.fss.2013.05.004.
[43] M. H. Asyali, D. Colak, O. Demirkaya, and M. S. Inan, "Gene expression profile classification: a review," *Current Bioinformatics*, vol. 1, no. 1, pp. 55–73, 2006, doi:0.2174/157489306775330615.
[44] H. Sarmadi, A. Entezami, B. Saeedi Razavi, and K.-V. Yuen, "Ensemble learning-based structural health monitoring by mahalanobis distance metrics," *Structural Control and Health Monitoring*, vol. 28, no. 2, p. e2663, 2021, doi:10.1002/stc.2663.
[45] K. Worden, G. Manson, and D. Allman, "Experimental validation of a structural health monitoring methodology: Part i. novelty detection on a laboratory structure," *Journal of sound and vibration*, vol. 259, no. 2, pp. 323–343, 2003, doi:10.1006/jsvi.2002.5168.
[46] E. Patel and D. S. Kushwaha, "Clustering cloud workloads: K-means vs gaussian mixture model," *Procedia computer science*, vol. 171, pp. 158–167, 2020, doi:10.1016/j.procs.2020.04.017.
[47] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society: series B (methodological)*, vol. 39, no. 1, pp. 1–22, 1977, doi:10.1111/j.2517-6161.1977.tb01600.x.
[48] G. McLachlan, "Finite mixture models," *A wiley-interscience publication*, 2000.
[49] D. Deng, "Dbscan clustering algorithm based on density," in *2020 7th international forum on electrical engineering and automation (IFEEA)*. IEEE, 2020, pp. 949–953, doi:10.1109/IFEEA51475.2020.00199.
[50] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE journal*, vol. 37, no. 2, pp. 233–243, 1991, doi:10.1002/aic.690370209.
[51] T. Kurita, "Principal component analysis (pca)," *Computer vision: a reference guide*, pp. 1–4, 2019, doi:10.1007/978-3-030-03243-2$_6$49 – 1.
[52] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010, doi:10.1002/wics.101.
[53] I. Goodfellow, "Deep learning," 2016.
[54] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," *Machine learning for data science handbook: data mining and knowledge discovery handbook*, pp. 353–374, 2023, doi:10.1007/978-3-031-24628-9$_1$6.
[55] InRete Lab, "Optical failure dataset," Scuola Superiore Sant'Anna, 2021, [Online]. Available: https://github.com/Network-AndServices/optical-failure-dataset.
[56] H. Junninen, H. Niska, K. Tuppurainen, J. Ruuskanen, and M. Kolehmainen, "Methods for imputation of missing values in air quality data sets," *Atmospheric environment*, vol. 38, no. 18, pp. 2895–2907, 2004, doi:10.1016/j.atmosenv.2004.02.026.

VII. Biography Section

Ingrid Ramos, Undergraduate student in Computer Engineering at the Federal University of Pará. She is an IT-A scholarship holder at the Laboratory of Applied Electromagnetism (LEA), where she conducts research on Machine Learning applied to anomaly detection.



Andrei Ribeiro, PhD student in Electrical Engineering at the Graduate Program in Electrical Engineering (PPGEE) of the Federal University of Pará (UFPA). Master's degree in Electrical Engineering from PPGEE and Telecommunications Engineer from UFPA. He is a researcher at the Laboratory of Applied Electromagnetism (LEA) at UFPA, where he conducts research involving Deep Learning techniques for optical network management. He has over 3 years of experience in research projects applying Machine Learning to areas such as agriculture, healthcare, and optical networks. He worked for a short period at the Samsung RD Institute as a research and development apprentice.

Fabrício Lobato, Bachelor's degree in Computer Science from the University of the Amazon (2005), Full License in Mathematics from the State University of Pará (2004), Master's degree in Computer Science from the Federal University of Pará (2009), and PhD in Electrical Engineering from the Federal University of Pará (2019). He was a tenured professor at the Federal University of Western Pará until 2021 and is currently an Associate Professor I at the Faculty of Computer Engineering and Telecommunications of the Institute of Technology at the Federal University of Pará (UFPA).

Moisés F. Silva Received the Ph.D. degree in electrical engineering from the Federal University of Pará, Brazil, in 2020. Currently a postdoctoral research associate at the Los Alamos National Laboratory (LANL), as part of the Mechanical and Thermal Engineering (E1) Group, working on the development of algorithms for inverse neural rendering and neuromorphic imaging for photo-realistic 3D modeling of scenes and monitoring of highspeed physical phenomena. Before joining LANL he has been a postdoctoral fellow at the Scuola Superiore Sant'Anna, Italy, for one and a half years performing research activities related to intelligent software-defined optical networks, internet of things, and network automation. Moises has collaborated in more than 30 papers published in peer-reviewed international journals and over 30 publications in national and international conference proceedings, currently serving as reviewer for a few international journals. At LANL, he has served as co-mentor for students enrolled in the Los Alamos Dynamics Summer School for a few years, leading to papers published in conference proceedings. Also, while at LANL, he has been granted a conference award for his work involving an LDRD project to develop an imager-based motion identification technique from dynamic point clouds using blind source separation and motion magnification techniques.

João C. W. A. Costa Holds a degree in Electronic Engineering from the Federal University of Pará (1981), a Master's degree in Electrical Engineering - Telecommunications from the Pontifical Catholic University of Rio de Janeiro (1989), and a PhD in Electrical Engineering - Telecommunications from the State University of Campinas (1994). He worked as a telecommunications engineer at DENTEL (ANATEL) and FUNTELPA between 1981 and 1986, developing installation feasibility projects and participating in the implementation of various sound and image broadcasting systems, including FM-Cultura, TV-Cultura, and the proposal for the new Basic TV Channel Plan for the State of Pará. Since 1994, he has been a faculty member at UFPA and became a full professor in 2018, working at the Faculty of Computer Engineering and Telecommunications at the Institute of Technology. He is a permanent researcher at the Graduate Program in Electrical Engineering and contributed to the creation of telecommunications and applied computing areas. He was the coordinator of the PPGEE (Oct 2004 - Dec 2005), Director of Research, and coordinator of the PIBIC at UFPA. He led the creation of the Graduate Program in Computer Science and was a permanent professor until 2023. He served as Deputy Secretary of Development, Science, and Technology - SEDECT (Jan 2007 - Dec 2010) and participated in structuring the CT system in the State of Pará. He was the President of the Brazilian Society of Microwaves and Optoelectronics (2012-2014), Vice-President (2010-2012), and is a member of the Brazilian Telecommunications Society and IEEE. A CNPq researcher since 1994, he was the first pro tempore Vice-Rector of the Federal University of South and Southeast Pará from August 2013 to February 2016. His research focuses on communication networks, applied electromagnetism, and applied computing. He has supervised more than a hundred graduate and postgraduate students, and he is a co-author of five international patent applications. He is a founding member and the first president of iSACI (Institute for Sustainability of the Amazon with Science and Innovation) and, since January 2025, has been the president of the Guamá Foundation for Science, Technology, Innovation, and Sustainable Development.