JOURNAL OF
Optical Communications
and Networking

Machine-learning-based technique to establish ASE or Kerr impairment dominance in optical transmission

Isaia Andrenacci,[1,2,*] Matteo Lonardi,[3] Petros Ramantanis,[1] Élie Awwad,[2] Ekhiñe Irurozki,[2] Stephan Clémençon,[2] and Sylvain Almonacil[1]

[1]Nokia Bell Labs, 12 Rue Jean Bart, 91300 Massy, France
[2]Télécom Paris, 19 Pl. Marguerite Perey, 91120 Palaiseau, France
[3]Nokia Bell Labs, Via Energy Park, 14, 20871 Torri Bianche, Monza and Brianza, Italy
*isaia.andrenacci@nokia.com

Received 26 September 2023; revised 29 January 2024; accepted 29 January 2024; published 18 March 2024

Data extraction from optical networks has increased substantially with the evolution of monitoring and telemetry methods. Using data analysis and machine learning, this paper aims to derive insights from this data, contributing to the development of self-optimized optical networks. More particularly, it focuses on predicting the Kerr and amplified spontaneous emission dominance by examining the fluctuations in the signal-to-noise ratio due to polarization-dependent loss. Building on previous work, which used the SNR statistic as the input feature of machine learning, our primary goal is to enhance prediction precision while concurrently decreasing the computational model's complexity. After refining the selection parameters of the input features, we observed a 70% reduction in the input feature length with respect to our previous work. The model reached a 98% accuracy rate, and it was able to successfully classify the regimes in a limited set of unseen experimental instances.

https://doi.org/10.1364/JOCN.506931

1. INTRODUCTION

In today's fast-paced world of communication, the need for cost-effective, high-speed, and reliable connections is ever-increasing. The traditional approach to deploying a new optical network leans heavily towards a "set-and-forget" mentality. This method involves designing network links with a focus on worst-case scenarios, often incorporating huge safety margins to guarantee seamless operation throughout the network's lifespan. While this cautious approach has historically served its purpose, it is increasingly being viewed as a suboptimal use of valuable resources. This is particularly true when considering the growing demand for higher network capacity and the need for improved operational efficiency in future optical networks.

Elastic optical networks (EONs) have recently emerged as a promising and innovative solution to tackle the challenges posed by modern communication requirements [1]. EONs capitalize on the flexibility offered by the new generation of transponders, enabling intelligent transmission of signals at higher capacities while seamlessly adapting to diverse traffic load scenarios and line constraints. This hardware resource diversification and flexibility brings a significant challenge to the optical communication control system. However, the dynamic and efficient management of an optical network requires an accurate representation of the optical network itself at all times.

To address this challenge, a solution is to implement a digital twin (DT) for control management. A DT refers to a high-fidelity, real-time, digital copy of an entity that closely mimics its real-world counterpart [2,3]. The aim of the DT is to construct an optical communication management system capable of adapting to the time-varying network environment. This idea is in opposition to the "set-and-forget" approach, for which worst-case degradations are assumed. This adaption ability of the DT is made possible by iteratively optimizing the virtual copy through real-time models. Then, the chosen optimization strategy is applied to the optical communication network. The DT may be leveraged to perform various functions such as hardware reconfigurations, transmission emulation, or failure prediction. In this work, we focus on the problem of identifying the type of noise dominating in point-to-point optical links, i.e., amplified spontaneous emission (ASE), or nonlinear noise generated by Kerr effects.

The ability to identify whether the system is operating in the linear or nonlinear regime is crucial for optimizing launch power and consequently enhancing the signal-to-noise ratio (SNR). This could be achieved by monitoring the optical

power of a transmission system. For instance, the longitudinal profile can be typically measured by employing optical time-domain reflectometers (OTDRs) or methods proposed in [4–6], thus providing a complete evolution of the optical power along the link. Nevertheless, the extra hardware or the additional complexity as sociated with the aforementioned methods may be prohibitive, while it is also expected to limit the ability of the DT to adjust in real time.

Artificial intelligence has arisen in recent years to build a DT for optical networks because it provides low-complexity real-time models without losing their accuracy. These characteristics are possible thanks to training, which moves the complexity in an initial phase before using the model. For instance, in [7], an artificial neural network (ANN)-based model was introduced to estimate nonlinear noise in live network scenarios. The proposed model demonstrated robust training in larger parameter spaces, encompassing link and transmitter parameters, resulting in low prediction errors across a wide range of configurations investigated. In contrast to this approach, a novel and monitoring fusion strategy is initiated to investigate the incorporation of monitored information into the model. This approach aims to achieve a more accurate estimation of performance, enhancing the model's ability to reflect real-world scenarios. Building on this concept, in [8], the authors used an ANN to predict how much Kerr nonlinear and linear ASE contributes to the SNR. They relied on measurements of the constellation diagram clouds, the amplitude noise covariance of the received symbols, and the number of channels. In a similar approach an ANN is proposed in [9] to predict the nonlinear SNR using the amplitude noise covariance of received symbols and network parameters. In [10], the authors used a long short-term memory network (LSTM) to predict the nonlinear power and SNR using the fast Fourier transform of the received signal. In a more recent study conducted by [11], a novel approach was introduced, leveraging the statistics of the bit error rate (BER) as input features using a fast BER histogram method for the estimation of the probability density function (PDF) of BER samples. Specifically, the authors employed an ANN to predict the linear and nonlinear noise-to-signal ratio (NSR) by utilizing the statistics of the BER, along with additional network parameters such as the number of channels and spans.

However, all previous proposals require using as input features various link and transmission parameters, such as symbol rate, link length, and the number of channels and spans, potentially in addition to performance statistics monitored at the receiver. Furthermore, all previous methods effectively perform regression on the exact values of the nonlinear and linear SNR. In contrast, our approach a) assumes solely the performance timeseries monitored at the receiver; b) aims to solve the problem through a less complex classification task that identifies the operational regime, rather than identifying the optimal power through regression; and c) through a specific normalization of the SNR PDFs we choose to concentrate solely on the shape of the distributions, in contrast to the actual PDFs. Furthermore, in this research, we harness the influence of polarization-dependent loss (PDL) within optical links. PDL is a phenomenon where the attenuation or loss of light varies depending on its polarization orientation. Although conventional optical fibers typically exhibit minimal PDL effects, it is crucial to acknowledge that certain optical components commonly utilized in terrestrial optical communication systems can introduce notable PDL. Specifically, devices like wavelength-selective switches (WSSs) integrated in reconfigurable optical add-drop multiplexers (ROADMs), and erbium-doped fiber amplifiers (EDFAs) can manifest distinct interactions with light of different polarization orientations. The presence of PDL in optical communication systems generally leads to fluctuations in the SNR of the received optical signal; it may induce crosstalk between the polarization tributaries and/or cause an unequal loss of signal energy, resulting in an imbalance in the performance of the two polarization tributaries. This power imbalance together with the random state of polarization rotations occurring in the fiber translates into oscillations of the BER over time [12]. Even though PDL has a detrimental effect, in this work we propose to leverage its presence to perform low-cost and low-complexity monitoring. Here, we build upon the foundation laid out in [13], where a machine learning (ML) classifier was introduced for monitoring the operational regime in optical links ("linear" or "nonlinear") using a statistical analysis of the SNR obtained from the received signal. Specifically, a two-step input feature preprocessing was applied, involving normalization and binning of the SNR distribution. This preprocessing emphasizes the PDF shape by effectively eliminating dependence on average SNR values. Hence, it enhances the generalizability of our technique beyond specific trained configurations. In [13], a k-nearest neighbors (KNN) algorithm with a fixed set of hyperparameters was both trained and tested by using a dataset obtained with the enhanced Gaussian noise (EGN) model, adapted to include PDL [12]. Extending the findings of [13], we delve into the exploration of four distinct ML classification models—KNN, support vector machine (SVM), ANN, and random forest (RF)—all leveraging the same EGN dataset. Our approach underscores the efficacy of diverse algorithms in addressing the problem, highlighting their higher computational efficiency and greater interpretability compared to the more common ANN [14]. Finally, we are using the proposed algorithms trained with the EGN dataset to identify the operating regimes in the case of limited unseen experimental data acquired with a different setup.

In Section 2, we review our methodology and data collection strategy of [13]. Unlike [13], we focused on the different ways of defining input features. We also present the fine-tuning of the four ML models under investigation. In Section 3, we present numerical results obtained from the training and fine-tuning of four distinct models. These models exhibited proficiency in recognizing both nonlinear and linear dominance, demonstrating their versatility without relying on assumptions about the knowledge of link and transmitter parameters. Our work illustrates that all four models consistently achieve an accuracy rate exceeding 97% in the test phase. This level of accuracy is comparable to the previous studies, but noteworthy is our ability to achieve the same accuracy with a 70% shorter input feature vector, thereby reducing the complexity of our ML models. Additionally, we conducted a comprehensive analysis of misclassifications to indicate the scenarios where our models might fall short. In Section 4, our

study deviates from prior research by introducing experimental validation in novel configuration scenarios that differ from the training datasets. This includes variations in fiber types, modulation formats, symbol rates, and the numbers of channels and spans. The experimental investigation demonstrates that these models are robust and effective even over the configurations that are unseen and not trained for. Finally, in Section 5, we wrap up our paper with a conclusion.

2. METHODOLOGY, SYSTEM SETUP, AND MACHINE LEARNING DETAILS

In this section, we first describe the methodology used to classify the operational regimes. Then, we provide the details of the emulated system. Finally, we describe the steps of data-processing and fine-tuning of the considered ML algorithms.

A. Methodology

For a particular set of point-to-point transmission systems, we used the EGN model, which incorporates the impact of the PDL. A large number of SNR samples were generated for different random states of polarization rotations along the line. From these SNR samples, we generate the PDFs. We repeated these steps for different input optical power levels corresponding to system operation in different regimes, spanning from weakly linear up to highly nonlinear, including the power level yielding the optimal performance denoted as the nonlinear threshold (NLT).

Then, we convert each of the aforementioned PDFs into a vector of input features, which are suitable for use with a supervised ML classifier, following a normalization procedure, which will be detailed later. We labeled each input feature vector according to the system operation regime which is set to be either "linear" if the power is below the NLT or "nonlinear" otherwise. Finally, we used these labeled datasets to train and test four ML algorithms.

B. EGN Simulation Setup for Collecting the SNR PDFs

In Fig. 1, we present the simulation setup, where the signal propagation occurred iteratively over spans, each comprising a standard single-mode fiber (SSMF) and an EDFA with 5 dB of noise figure. The fiber's dispersion parameters were set at 16.7 ps/nm/km, and each span had a length of 100 km, an attenuation of 0.22 dB/km, and a nonlinear parameter of $1.26\,\mathrm{W}^{-1}\,\mathrm{km}^{-1}$ at 1550 nm. One ROADM was placed after the transmitter and another one before the receiver, operating in the add or drop mode, acting as WSSs. Within the transmission line, the ROADMs were operated in bypass mode, involving two WSS elements, assumed not to add filtering penalties. We used an EGN model extended to PDL [12], for the simulation of the SNR fluctuations.

We emulated six distinct scenarios by varying the transceiver and line setups to evaluate the performance of the proposed technique under different conditions. The transceiver operated either at 49 GBd with 50 GHz spacing or at 69 GBd with 75 GHz spacing. In all scenarios, we considered 21 channels with Gaussian modulation and analyzed the performance of the central channel.

Along the transmission line, we explored different numbers and locations of the ROADM, as well as different probability distributions for selecting PDL values of the WSSs in the setup. We investigated two cases for the ROADM location, depicted as the yellow switch in Fig. 1: first, the regular pattern where a ROADM was cascaded after a series of three spans, and second, the random pattern where a ROADM was randomly inserted at each span with a 30% likelihood. Regarding the PDL-generation mechanism, two options were considered for introducing PDL values. The first approach involved randomly selecting PDL values for each PDL element from a uniform (U) distribution ranging between 0.1 and 1 dB. The second option utilized a chi-square (χ^2) distribution with three degrees of freedom, with a mean of 0.2 dB and a probability of exceeding 0.8 dB set at 1.05%. For each scenario, we investigated 20 different random realizations of PDL elements.

To assess the impact of different lightpath lengths, we considered four options: 12, 15, 18, and 21 spans, as indicated by the blue switch in Fig. 1. Additionally, we varied the transmitted power P_{launch} from −10 to 10 dBm in steps of 0.5 dB. In total, we collected SNR fluctuations from six different scenarios, each involving specific parameters related to PDL generation, ROADM pattern, and symbol rate.

We gathered a total of 1 million SNR samples for each simulation run. Subsequently, we computed the corresponding PDFs of the SNR samples for each scenario. Figure 2 shows an illustrative example of the collected SNR PDFs. Specifically, we report the SNR PDFs for 49 GBd signals and a ROADM

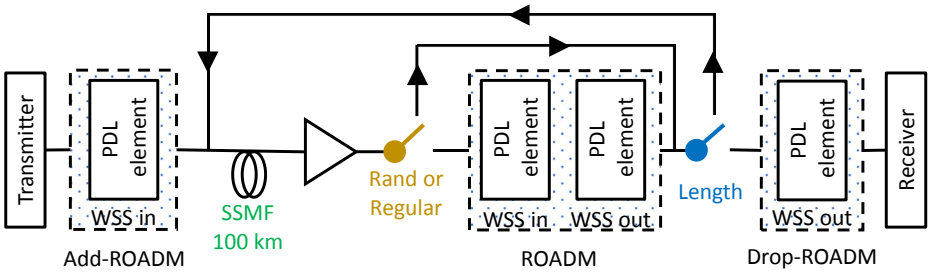

Fig. 1. Simulation setups used to collect the SNR samples. At the transmitter, different symbol rates and channel spacings were employed in a fixed 21-grid with Gaussian modulation. The optical link consists of a repetition of an SSMF + EDFA span, followed by either a "regular" or "random" location of the ROADM (two WSS cascade). PDL elements are independently chosen using chi-squared (χ^2) or uniform (U) distributions. Specifically, links composed of 12, 15, and 21 spans were investigated.

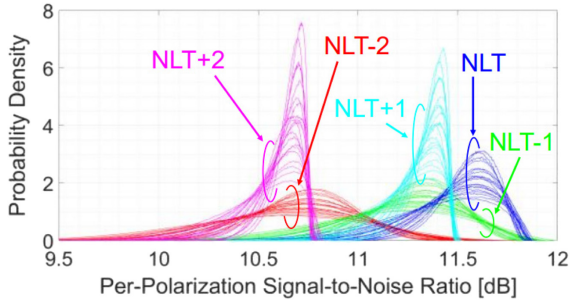

Fig. 2. Examples of SNR distributions for optical signals transmitted at 49 GBd through a 21-span link with a regular ROADM pattern, i.e., a ROADM after every 3 spans. Each line on the plot represents a distinct PDL sequence realization, while different colors indicate various power levels ranging from $P_{\text{launch}} = \text{NLT} - 2$ (linear) dBm to $P_{\text{launch}} = \text{NLT} + 2$ (nonlinear) dBm.

every 3 spans, within a total length of 21 spans. The analysis considers five different power levels located near the NLT. For each power, we conducted 20 separate realizations to capture the variability of the SNR PDFs.

In this study, it is essential to recognize that, while in the linear regime distributions are Gaussian-like symmetric, SNR PDFs in the nonlinear regime exhibit asymmetry due to the fact that NLI-PDL interaction is fundamentally different from the interaction between ASE and PDL. This phenomenon has been experimentally observed and numerically validated in [15], while a theoretical justification and discussion may be found, e.g., in [12] (Section III). This characteristic holds considerable importance for the proposed ML classification, enabling the differentiation between these regimes based solely on SNR observations.

C. Data Preparation

1. Input Feature Extraction

We proposed a two-step procedure to extract the necessary input features. Figure 3 depicts a qualitative example of an asymmetrical nonlinear distribution. The first step, presented at the top of Fig. 3, involves normalizing the PDF of the SNR.

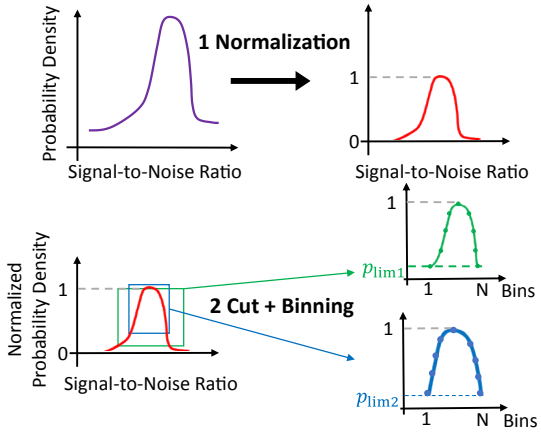

Fig. 3. Two-step procedure used to extract input features in [7] at fixed p_{lim} and N values. The left part represents the normalization of the SNR distribution, while the right part showcases the second step, involving cutting and binning at fixed p_{lim} and N values with two distinct threshold values $p_{\text{lim2}} > p_{\text{lim1}}$.

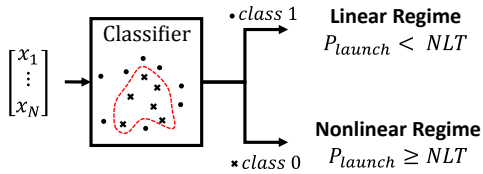

Fig. 4. Sketch of the classification problem where the input feature vector representing SNR distribution is used to predict the two target classes, i.e., linear and nonlinear regimes.

Normalization in this context refers to rescaling the SNR PDF values between 0 and 1. The second step, illustrated at the bottom of Fig. 3, consists of two consecutive processes: cutting and binning the normalized PDF of the SNR. The "cut" is executed using a fixed threshold p_{lim}, where only the data points above p_{lim} are kept, and the "binning" is performed by dividing the remaining data into a fixed number of bins N. Hereafter, p_{lim} and N will be referred to as input feature parameters. Finally, we extracted an N length input feature $[x_1, \ldots, x_N]$ for each of the collected SNR PDFs.

2. Data Labeling

We have established a classification system comprising two distinct classes: the "linear regime" for scenarios in which the launch power (P_{launch}) falls below the NLT and the "nonlinear regime" for all the launch power exceeding this threshold. The NLT, which signifies the launch power that optimizes the SNR [16], has been evaluated for each investigated link without considering PDL. We labeled each input feature vector obtained in the previous step with these two classes. Figure 4 illustrates this binary classification problem, with class 1 representing the linear regime and class 0 representing the nonlinear regime.

At fixed input feature parameters, we generated a labeled dataset of 20,000 instances, including all six scenarios. Unlike the approach taken in [13], where a single labeled dataset was generated with a fixed $p_{\text{lim}} = 10^{-3}$ and $N = 100$; in this work, we generated multiple labeled databases by varying the threshold limit and the number of bins. This approach allows for a more comprehensive analysis of how varying p_{lim} and N influences the ML model's performance.

3. Data Analysis

We examine the effects of the simulation setup parameters, threshold, and number of bins on the input feature vectors collected in the previous subsection.

The threshold limit plays a crucial role in shaping the tails of the SNR distributions, representing rare events such as extremely high or low SNR values for a fixed configuration setup. Accurately estimating these tails poses a challenge due to the limited availability of SNR samples [17]. For a fixed number of bins, employing higher threshold values provides a more precise depiction of the SNR distribution; however, this precision comes at the cost of losing information from the extreme tails. Conversely, using lower threshold values considers the noisy estimated values of the PDF but can result in a reduction in the accuracy of the classification model. To grasp the idea of this trade-off, consider the qualitative graph at

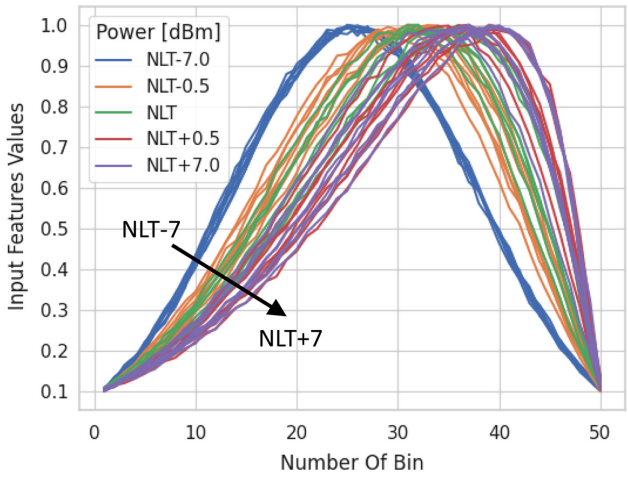

Fig. 5. Examples of input feature vectors (normalized SNR PDFs) for a fixed probability threshold and a bin number, respectively 0.1 and 50. The figure represents different transmitted powers with a fixed number of spans of 18. For each of these powers, six realizations are reported with the same color to represent the different scenarios.

the bottom side of Fig. 3. This figure represents an example of a normalized PDF in the nonlinear case. Two threshold values p_{lim1} and p_{lim2} are compared with the first much smaller than the second. The resulting two input feature vectors are the zoomed version of the normalized probability density. Note that, in the case of a smaller threshold limit, the asymmetric characteristic of the nonlinear probability density is still visible. Therefore, a specific threshold value should exist that permits a more accurate normalized SNR distribution while capturing the tilting of the nonlinear case.

A higher number of bins allows for a more accurate representation of the normalized probability density shape. However, this increases the input feature length, leading to higher computational complexity for the ML algorithms. Hence, there is a trade-off between accuracy and ML model complexity.

Although we have not applied any supervised model yet, we can make some observations on the input features, which will help to gain insight into the forthcoming results. As an example, consider Fig. 5, where several realizations of the input features are represented. These realizations are drawn from the labeled dataset with $p_{\text{lim}} = 0.1$ and $N = 50$. These input features are reported for five different powers, respectively, NLT $-$ 7, NLT $-$ 0.5, NLT, NLT $+$ 0.5, and NLT $+$ 7. Meanwhile, the number of spans was kept fixed at 18. This distance is sufficient to generate high accumulated nonlinearity. We reported six different realizations with the same color for all these investigated powers. Each of these realizations represents one of the six scenarios described above (one out of two symbol rates, one out of two ROADM location strategies, and one out of two PDL distributions).

We first note that the six considered scenarios yield very similar input feature vectors (normalized SNR PDFs), therefore advocating for the robustness of the explored method. Second, we note that the input features near the NLT are very similar, while the linear and nonlinear regimes have very distinct shapes.

Given our definition of two regimes in our dataset, it is worth noting that there is a slight imbalance in the label or class distribution in our dataset. Approximately 60% of the instances fall into the linear regime category, while the remaining 40% belong to the nonlinear regime. Despite this minor disparity, we have opted to utilize accuracy as the primary performance metric for our ML models, as elaborated in the following section.

Furthermore, we observed that the constructed labeled datasets have a uniform distribution of the transmitted power and span number for both linear and nonlinear cases. This uniform distribution ensures that the ML model encounters instances from various power levels and spans with the same probability during training and testing. This distribution is beneficial for training the ML models as they learn from all possible power and span combinations.

D. ML Model Tuning and Performance Evaluation

As a difference from [13], we assessed the classification problem with four different algorithms and compared their performance. Specifically, we focused on KNN, SVM, ANN, and RF. We review these four ML models and their used hyperparameters in Appendix A.

In this subsection, we explain how we evaluate and compare the performance of each classifier. First, we divided the 20,000 instances into training and test sets for each dataset at a fixed threshold limit and a bin number. Specifically, we split each dataset into an 80% for training and tuning and a 20% for testing.

1. *Training: K-Fold Cross-Validation for Model Tuning*

The process of evaluating and comparing the performance of each classifier through cross-validation is essential to tune hyperparameters and, in this work, the input feature parameters of each ML model. In this work, we explore different hyperparameters reported in Table 1 for tuning the four ML models.

To begin, we focus on one fixed dataset, but the following procedures apply to the other datasets as well. The training set is divided into K multiple subsets or folds to perform K-fold cross-validation. During cross-validation, K $-$ 1 folds are used for training, and the remaining one is used for validation. This process is repeated K times to ensure that every data point is used for both training and validation. As a result, we obtain K different values of accuracy, one for each fold. The average performance over K iterations provides a more reliable estimation of the model's test performance, and it helps mitigate the risk of overfitting. Additionally, calculating the standard deviation or other measures of variance from the K accuracy values indicates the model's stability and consistency in performance.

To evaluate the classification performance, we use accuracy as the performance metric. This metric measures the proportion of correctly classified instances out of the total number of instances. Accuracy is chosen because the class distribution is relatively balanced, as shown in the previous subsection, and there is no distinction in terms of importance between the two classes (nonlinear and linear regimes).

2. Testing: Misclassifications Analysis

After completing the tuning stage, we proceed with the test phase and error analysis of the four ML models. Using the unseen test data, we evaluate the accuracy of the four tuned ML models. The misclassifications are carefully analyzed to understand the impact of different simulation parameters, such as the transmitted power and link length, on the ML models' performance when errors occur.

3. RESULTS: ML APPLIED TO SIMULATION DATA

In this section, we present the results of the impact of different probability limits and the number of bins on the estimated test accuracy, i.e., cross-validation accuracy. Then, we report the test accuracy of the four tuned ML models with an analysis of their misclassifications.

A. Training Results: 10-Fold Cross-Validation for Model Tuning

1. k-Nearest Neighbors

For KNN we perform a comprehensive grid search, by exploring different hyperparameters such as the number of neighbors and distance metrics, using various labeled datasets with distinct input feature parameters. Among the combinations, we achieved the highest accuracy when employing a threshold limit of 0.3, using 70 bins for feature representation, setting k to 1, and utilizing the L1 metric for the KNN model. Upon analyzing the hyperparameter variations, we observed that the accuracy values remained relatively stable across different distance metrics, except for Chebyshev [18], which showed slightly lower accuracy. Additionally, the choice of the number of neighbors did not have a substantial impact on the test's estimated accuracy, with the optimal performance observed at $k = 1$.

Figure 6 illustrates the cross-validation accuracy of the KNN model as a function of the threshold limit and number of bins.

Table 1. Different Hyperparameters Investigated in the Four Used ML Models, Where *N* Is the Input Feature Length

	Hyperparameters		Values
KNN	Metric		L2, L1, cosine, Hamming, Chebyshev
	K neighbors		1, 3, 5, 7, 9, 11, 13
RF	Tree depths		1, 3, 5, 7, no constraints
	Number of trees		100, 200, 500
	Percentages for bootstrapping		10%, 40%, 60%, 80%
	Feature for splitting		sqrt(N), N/2, N
ANN	Activation functions		Logistic, relu, tanh
	Number of neurons		5, 10, 15, 20, 25
	Number of hidden layers		1, 2
SVM	Radial kernel	Gamma	0.1, 1, 10
	Polynomial kernel	Degree	1, 2, 3, 4, 5, 6
	C value		0.01, 0.1, 1, 10, 100

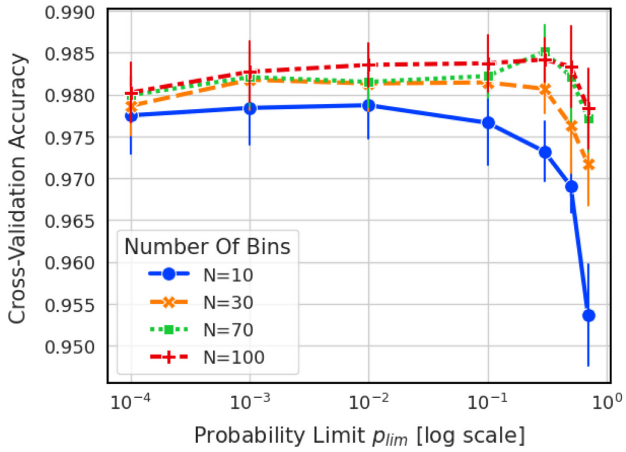

Fig. 6. KNN cross-validation accuracy as a function of threshold limits used to extract the input feature. The lines represent the accuracy for different values of the number of bins.

Each line in the plot corresponds to a different number of bins used in the analysis. The markers represent the average accuracy, while the error bars indicate 2 times the standard deviation of the 10 accuracy values obtained from the cross-validation technique. The figure shows that the KNN model's cross-validation accuracy remains relatively stable across various threshold limits, with a slight decline observed after the probability limit of 0.3, which coincides with the point of achieving the maximum estimated accuracy.

This behavior confirms the intuitions discussed above about the threshold limit. A high value of p_{lim} distorts the PDFs, leading to a lower accuracy. On the contrary, a low threshold value preserves the PDF shape, thus resulting in a high accuracy. Furthermore, the figure shows the result for different bin values, which reach similar accuracy values for bins above 10. Therefore, the accuracy is minimally affected by fluctuations in the number of bins. This observation is supported by the overlapping error bars for different bin values, suggesting that the KNN model's performance is relatively insensitive to the number of bins used for feature representation.

2. Random Forest

For the RF model, we conducted the same procedure as with the KNN model to determine the optimum hyperparameters and input feature parameters. Through 10-fold cross-validation on all labeled datasets, we found that the RF model had the best accuracy with a threshold limit of 0.01, by using 100 bins. Additionally, we found that setting 80% of the dataset for bootstrapping yielded the best results, and there were no constraints on the tree depth. For the number of trees in the ensemble, the optimum value was 500. Moreover, we fixed the maximum feature for splitting to the square root of the input feature length, i.e., the number of bins. An interesting observation was that the tree depth was the only hyperparameter that had a significant impact on the model's performance accuracy. When the tree depth was set to be unconstrained, it led to an improvement of approximately 2% in accuracy compared to using a tree depth of 1. This indicates

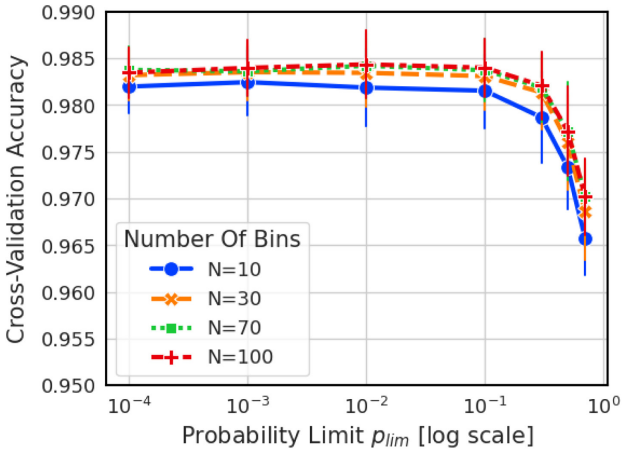

Fig. 7. RF cross-validation accuracy as a function of probability limits used to extract the input feature. The lines represent the accuracy for different values of the number of bins.

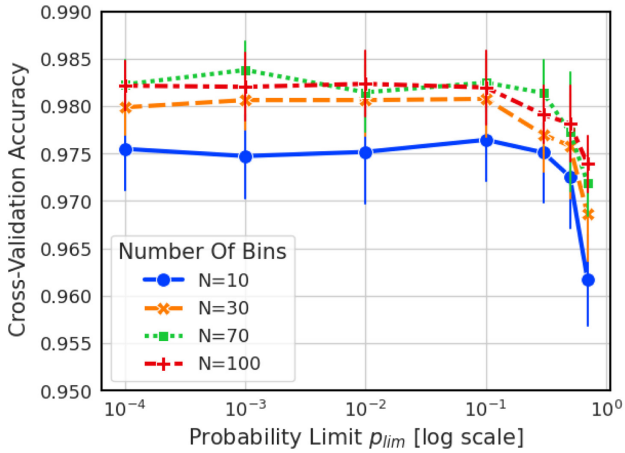

Fig. 8. Cross-validation accuracy as a function of probability limits for the SVM model. The lines represent the accuracy for different values of the number of bins.

that allowing the trees to grow deeper slightly improved the model performance.

With these fixed optimum hyperparameters, we further investigated the impact of input feature parameters on the cross-validation accuracy for the RF model, as shown in Fig. 7. The behavior of accuracy was quite like what was previously observed with the KNN model. However, this time, we noticed that the probability limit started to decrease after the threshold of 0.1, suggesting that the model's decision-making became more sensitive to changes in the probability limit within this range.

Furthermore, the difference in accuracy observed earlier for the KNN model concerning the number of bins became less pronounced in the case of the RF model. This indicates that the RF model's performance was less affected by variations in the number of bins used for feature representation compared to the KNN model.

3. Support Vector Machine

We repeated the cross-validation process for the SVM model using the hyperparameters specified in Table 1. We performed this cross-validation for each dataset with different probability limits and numbers of bins. In the SVM model, we found that the optimum performance was achieved with a third-degree polynomial kernel. Additionally, the best values for the hyperparameters were a C value of 10 and a gamma value of 0.1.

After determining these fixed optimum hyperparameters, we illustrate the SVM model's performances in Fig. 8. The cross-validation accuracy obtained from the SVM exhibited very similar behaviors with different input parameters. Like the RF model, the decreasing trend in accuracy started at probability limits around 0.1.

Interestingly, this model showed a more significant decrease in cross-validation accuracy when the number of bins was fixed at 10, compared to the other models previously investigated. This implies that the SVM model is more sensitive to a lower number of bins when it comes to decision-making.

4. Artificial Neural Network

For the last investigation, we conducted a cross-validation analysis employing a shallow ANN. Various neural network configurations were explored, encompassing a maximum of two hidden layers, each having a variable number of neurons, similar to [11]. The experimentation also encompassed three different activation functions, as detailed in Table 1.

After subjecting all labeled datasets to a 10-fold cross-validation process, we determined that the ANN model exhibited the highest accuracy when employing a threshold limit of 0.0001 and utilizing 70 bins. Using, the neural network configuration featuring two hidden layers of 10 neurons with the hyperbolic tangent (tanh) activation function produced the most favorable outcomes.

With the identified optimal hyperparameters, our investigation extended to examine the impact of input feature parameters on the cross-validation accuracy for the ANN model. Figure 9 presents a depiction of the cross-validation accuracy as a function of different probability limits, following the same methodology as employed in our analysis of previous models. Similar to the RF model, the number of bins utilized does not significantly affect cross-validation accuracy. Furthermore, similar to all three previous ML models, the ANN maintains a rather constant accuracy for probability limits lower than 0.1, nevertheless exhibiting a slightly lower average cross-validation accuracy overall.

5. Comparison of the Four ML Models

To gain a comprehensive understanding of the maximum achievable performance of the four different algorithms, we selected fixed optimum input feature parameters based on the previous results. We used a violin plot to observe the cross-validation performance of four ML models. A violin plot displays the 10-fold accuracy distribution by featuring a white dot for the median, and a thick gray bar for the interquartile range. A violin-shaped curve on each side depicts the probability distribution of the 10 accuracy values obtained from the cross-validation technique. In Fig. 10 we illustrate with this violin plot for the four investigated different ML models.

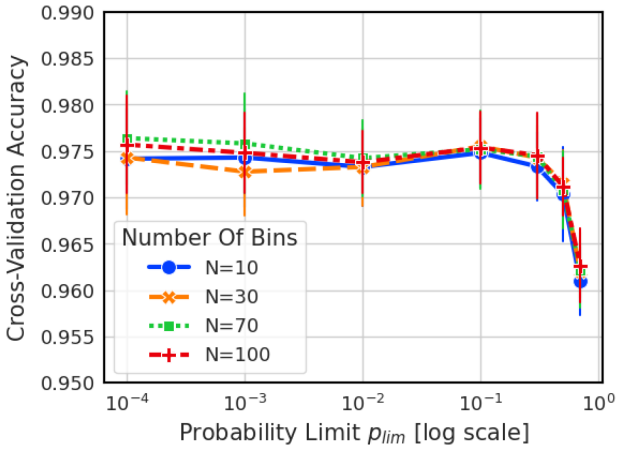

Fig. 9. Cross-validation accuracy for the ANN model as a function of probability limits. The lines depict the accuracy for different values of the number of bins.

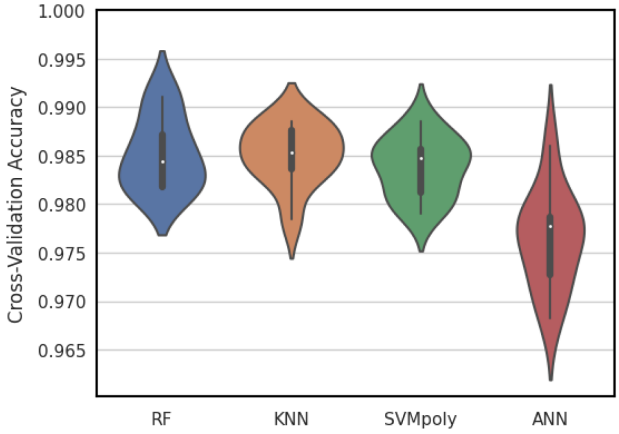

Fig. 10. Cross-validation accuracy distributions of four different algorithms, respectively, KNN, RF, SVM, and ANN.

The RF, KNN, and SVM models display similar performances and distributions. In contrast, the ANN model exhibits a lower median and a distribution with larger tails, reaching a minimum of 95.5%. Remarkably, the RF model showcases a distribution tilted toward higher accuracy, with worst-case scenarios surpassing those of the other models Moreover, the KNN model stands out with a higher median cross-validation accuracy. Overall, all the analyzed models achieve a median cross-validation accuracy above 97%. These findings indicate that the selected models demonstrate a highly promising average accuracy during the testing phase. Upon closer examination of Fig. 10, it becomes apparent that the RF model is the preferred choice, particularly in the worst-case scenario.

To quantitatively assess the complexity of the considered ML algorithms, we measured their corresponding training and testing times on a server with 540 Gb of memory using one CPU Intel Xeon Gold 6338N at 2.20 GHz. Results are reported in Table 2. Notably, the key takeaway from these results is that the most resource-intensive training is done by the RF, which takes about 10 s. On the other hand, testing is faster for the ANN compared to all other algorithms.

Table 2. Training and Test Time of the Four ML Models Investigated

	Training Time [seconds] (16,000 instances)	Test Time [seconds] (4000 instances)
KNN	0.0015 ± 0.0001	0.369 ± 0.018
RF	10.33 ± 0.041	0.062 ± 0.0002
ANN	1.985 ± 0.002	0.0012 ± 0.00001
SVM	1.0314 ± 0.002	0.0919 ± 0.0002

B. Testing: Misclassifications Analysis

Our goal is to find an optimum combination of input feature parameters while considering two constraints. First, we aimed to reduce the input feature length to decrease the ML complexity, thus selecting a lower number of bins. According to the previous analysis, we settled on using 30 bins since it was shown that higher values did not lead to improved performance. Second, we sought a higher probability limit to filter out unreliable tails in the data distribution, which represented improbable SNR values. These tails were particularly significant when dealing with experimental results since the experimental data has a less accurate tail estimation, as we will see in the next section. To achieve this, we chose a probability limit before the point of decreasing validation accuracy, which was 0.1. This allowed us to maintain high estimated test accuracy while eliminating noisy estimates from the tail of the distribution.

During the test phase, we evaluated the performance of all four ML models using 4000 unseen instances. The results revealed high accuracy for all models, with RF leading the way at 98.75%, followed by KNN at 98.52%, SVM at 98.17% and ANN at 97,68%. All the results have been succinctly summarized in the left column of Table 3.

To gain insights into the cause of misclassifications, we focused on analyzing the RF model, which achieved the highest accuracy. Among the 4000 test instances, we observed 49 misclassifications made by the RF model. Our objective was to identify the specific scenarios where the model tended to fail more often. In particular, we closely analyzed the launch power and the number of spans that caused the majority of these misclassifications. This analysis aimed to shed light on the situations in which the RF model might have encountered challenges and make further improvements to the model's performance.

By plotting a histogram of the percentage of misclassifications as a function of launch powers in Fig. 11, we observed the majority of misclassifications occurred at a power equal to

Table 3. Accuracy Test for the KNN, RF, SVM, and ANN Models at Fixed Optimum Input Feature Parameters

	Numerical Test Accuracy (4000 instances)	Experimental Test Accuracy (10 instances)
KNN	98.52% (58 misclassifications)	90% (1 misclassification)
RF	98.75% (49 misclassifications)	100% (0 misclassifications)
ANN	97,68% (91 misclassifications)	70% (3 misclassifications)
SVM	98.17% (72 misclassifications)	90% (1 misclassification)

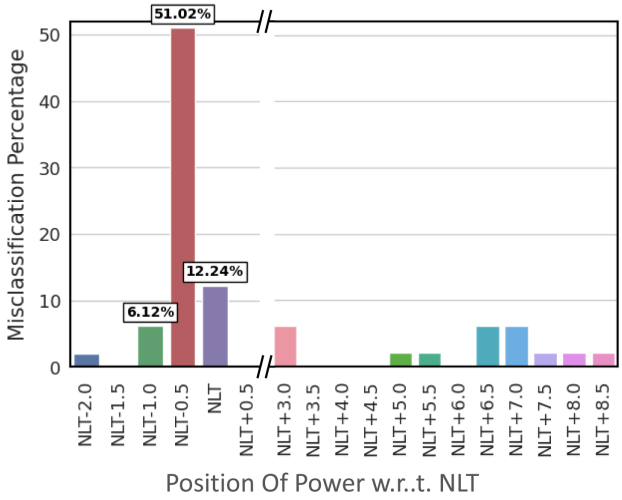

Fig. 11. Misclassification percentage of the RF model as a function of different transmitted power positions with respect to the NLT.

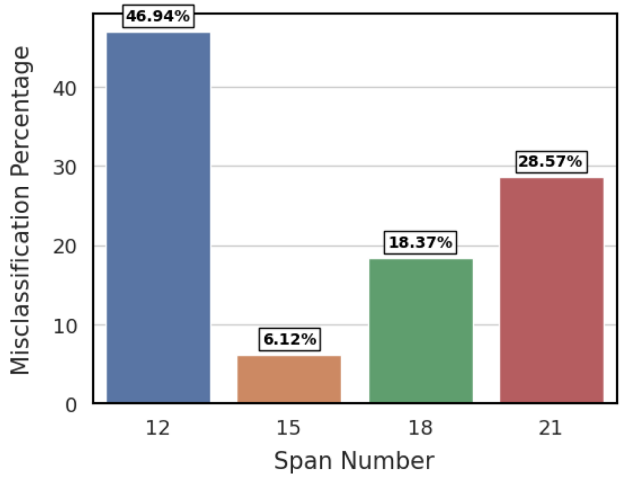

Fig. 12. Test misclassification percentage of the RF model as a function of different spans.

the NLT or near the NLT in the linear regime. In fact, approximately 70% of the misclassifications were attributed to input powers close to NLT − 0.5, NLT, and NLT − 1. However, note that misclassifications are generally rare events and their corresponding histograms (e.g., see NLT + 3 up to NLT + 6) are not very accurately assessed, and Fig. 11 should be therefore interpreted qualitatively and not quantitatively for these power levels.

This analysis indicated that the proposed ML model showed significant accuracy in identifying high linear or nonlinear regimes. However, it struggled when it came close to the NLT in the linear regime. This emphasized the importance of considering different transmitted power distributions for testing this ML classification technique in future works.

Furthermore, the misclassifications analysis was extended to include the KNN, SVM, and ANN algorithms. Interestingly, the conclusions drawn from the RF analysis also hold true for the KNN and ANN algorithms. Like the RF model, the KNN and ANN algorithms have many misclassifications in the linear regime and around the NLT value. Around 75% of misclassifications in the KNN model were linked to transmitted powers near the NLT in the linear regime. Similarly, 66% of misclassifications in the ANN model were associated with this range, underscoring the impact of transmitted power distribution on the classifier's performance. However, it is worth noting that the SVM algorithm exhibited a different behavior, with a misclassification portion occurring in the high nonlinear regime. The misclassification percentage in this nonlinear regime was around 20%, indicating that the SVM model struggled to accurately classify instances with high levels of nonlinearity.

We also investigated how different spans affect the misclassification error of ML classifiers. Figure 12 displays the misclassification percentage of the RF as we varied the number of spans (12, 15, 18, and 21). Our findings reveal a clear trend: smaller spans lead to higher error rates, with 46.94% of errors occurring with the 12-span scenario. This can be attributed to wider SNR distributions in both regimes, resulting from fewer accumulated PDL elements along the link. Interestingly, errors also seem to increase with larger spans. These consistent trends in misclassification errors were observed across all studied classifiers. Hence, we conclude that the PDFs corresponding to both higher and lower numbers of spans exhibit similarities, which leads to an error in the classification.

4. RESULTS: ML APPLIED TO EXPERIMENTAL DATA

In this subsection, we employ the same four tuned ML models (RF, KNN, SVM, and ANN) to classify the regions with experimental data. Based on the insights gained from the cross-validation process, we fixed the number of bins to 30 and set a probability limit of 0.1, as these parameter values consistently demonstrated the best performance. Consequently, we utilized these trained models to test them against unseen experimental scenarios.

A. Setup, Data, and Learning

In Fig. 13, the presented experimental data is derived from the testbed outlined in [19]. This configuration involves a comb of 13 channels operating at a symbol rate of 32.5 GBd with PDM-QPSK modulation, evenly space by 50 GHz. The transmitter employed three distinct power levels for experimentation: linear, nonlinear dominant, and at the optimum power. The comb signal traversed between 10 and 12 recirculating loops, where each loop corresponded to a pairing of an EDFA and 100 km of large effective area fiber (LEAF). The fiber's parameters included a chromatic dispersion of 4.3 ps/nm/km, an attenuation of 0.22 dB/km, and a nonlinear coefficient of 1.5 W^{-1} km^{-1} at 1550 nm. To emulate PDL, a polarization scrambler followed by a PDL element was utilized, calibrated offline with 1 dB of PDL. At the receiver, processes such as carrier phase recovery and equalization of linear impairments were executed. The SNR of each polarization tributary was estimated based on the variance of the constellation clouds. Approximately 1000 SNR samples were collected at two launch powers: 1) in the linear regime, approximately 4 dB below the NLT, and 2) in the nonlinear regime,

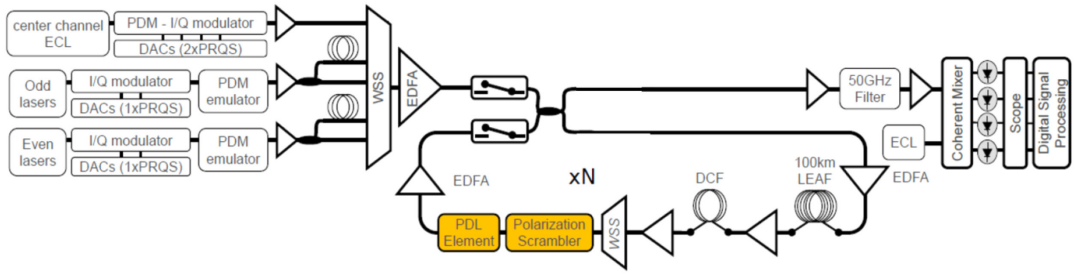

Fig. 13. Experimental setup for collecting the SNR fluctuations in [20].

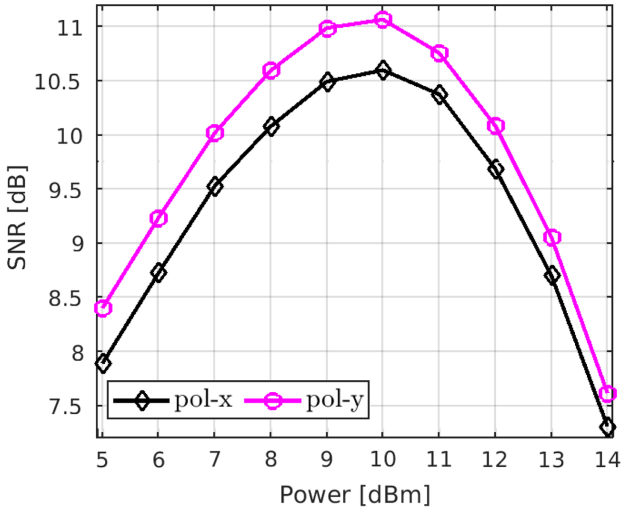

Fig. 14. Average per-polarization SNR as a function of different input powers. The average is carried out across five acquisitions in the experimental setup without a PDL element.

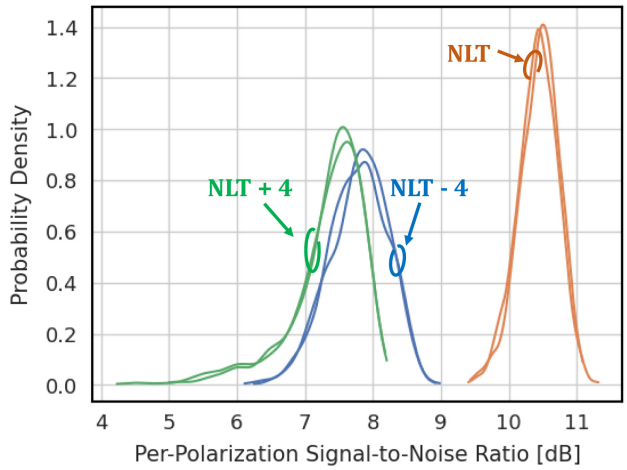

Fig. 15. Examples of experimental SNR PDFs. For a link of 12 spans, each line represents polarization, while different colors indicate various power levels.

approximately 4 dB above the NLT. This ensured that in each case, either ASE or nonlinear dominated the total variance. In this experiment, we determined the optimal launch power by conducting five repetitions of the experiment without the PDL element in the link. This procedure was repeated with different launch powers, ranging from 5 to 14 dBm in steps of 1 dBm. Figure 14 illustrates the average per-polarization SNR of these experiments as a function of different power levels. Hence, the NLT of the experimental setup was identified to be 10 dBm for both loops investigated. For the first set of 10 spans, we collected SNR sequences for two distinct transmitted power scenarios: linear and nonlinear. In the loop of 12 spans, the SNR sequences were collected for three different transmitted power scenarios: linear, nonlinear, and at the NLT. Hence, it was gathered in a total of 5 sets of SNR PDFs for each polarization, corresponding to a total of 10 instances.

Due to the relatively small sample size, the PDFs of the SNR appeared noisy; hence we applied interpolation to the PDFs of the SNR, resulting in smoother representations.

Figure 15 shows the interpolated SNR PDFs for the 12-loop experiment, showing three cases: linear, nonlinear, and at the NLT, each with two realizations corresponding to the two polarizations (x and y).

Observing the interpolated PDFs, we noted that the nonlinear PDFs exhibited a typical left tilt, while the linear PDFs followed Gaussian-like shapes, consistent with the patterns observed in the simulation results while the PDF corresponding to the NLT was centered around the highest SNR.

B. Results

We evaluated each model's performance based on accuracy and misclassifications. The RF classifier achieved the highest accuracy of 100%. On the other hand, both the KNN and SVM models achieved an accuracy of 90%, with one misclassification, while the ANN results in 70% with three misclassifications. Table 3 summarizes the test accuracy obtained with the numerical and experimental data.

The experimental setup differed from the simulation, but the ML models showed promising results in predicting unseen experimental scenarios. It is important to note that 8 out of 10 experimental data were in either the linear or nonlinear regime. In this region, we demonstrated that the four algorithms performed a correct classification of these SNR PDFs.

On the contrary, the SVM and KNN models were not able to correctly classify one of the SNR PDFs corresponding to the NLT. This finding was identified in the previous section, where errors were located, especially at transmitted powers in the linear regime near the NLT.

In conclusion, this section provides valuable insights into the performance of the ML models using real-world experimental data. The successful validation demonstrates the

models' capability to handle unseen real scenarios. However, it is crucial to highlight that these results are based on a limited experimental dataset. Therefore, a larger experimental dataset is needed to accurately validate and confirm these findings.

5. CONCLUSION

This paper presented a comprehensive investigation and extension of the ML-based solution proposed in [13] to classify optical communication systems operating in either a linear or nonlinear regime. The focus of this extension work was on fine-tuning the input feature parameters, namely, the number of bins and probability limits, and evaluating the classification performance of different ML models while considering possible complexity reductions. Furthermore, we gave a deeper understanding of the physical background, and we performed an experimental validation.

To gain a deeper understanding of the problem, we conducted a detailed data analysis, exploring the effects of various parameters in the simulation setup on the input feature representing the SNR distribution. This analysis was crucial in comprehensively examining misclassification patterns and understanding the impact of different simulation setup parameters on the SNR distribution.

We employed four ML models, namely, k-nearest neighbors, random forest, support vector machine, and artificial neural network, to perform the classification task. Through an in-depth analysis of the models' results with respect to the input feature parameters, we observed that validation accuracy remained relatively consistent for threshold limits below 0.1 across all four algorithms. Moreover, we found that the number of bins had minimal impact on validation accuracy once the value exceeded 10.

In the testing phase with simulation data, we achieved high accuracy above 97% for all ML models. However, a closer examination of misclassification errors revealed issues related to the transmitted power in the linear regime, especially near the launch power yielding the maximum SNR, as well as the impact of smaller spans. Recognizing these sources of errors has yielded valuable insights for enhancing future models and system performance.

Furthermore, we demonstrated the models' ability to predict outcomes in experimental scenarios that were previously unseen. Notably, the RF model achieved an accuracy of 100% in these scenarios, highlighting the potential of ML-based solutions for real-world applications.

In conclusion, our study underlines the significance of careful parameter tuning and data analysis in developing accurate ML classifiers for optical communication systems. Future work could explore further refinements to the models, leveraging the insights gained from this research to enhance classification performance and broaden the applicability of ML in optical communication systems with more realistic impairments including filtering penalties and transponder imperfections.

APPENDIX A: ML MODELS

In this appendix, we briefly review the ML models that played a pivotal role in our study: KNN, SVM, RF, and ANN. Moreover, we shed light on hyperparameters, which are parameters related to these ML models. These parameters are configured before the training process, influencing how the ML models operate. Properly tuning these hyperparameters is essential for optimizing the models to achieve their highest potential accuracy. For a more in-depth understanding of these models and their associated hyperparameters, we encourage you to consult our referenced sources [14,20].

1. k-Nearest Neighbors

KNN is a non-parametric classification algorithm that finds the k-nearest training instances (neighbors) to a given test instance in the feature space. It classifies the test instance based on the majority class among its k-nearest neighbors [14]. In this work, we varied the following hyperparameters.

- *k (integer):* the number of nearest neighbors to consider when making predictions. In binary classification, it is preferable to choose an odd number to avoid ties. Choosing the correct k value is essential, as smaller values (e.g., $k = 1$) can lead to overfitting, while larger (e.g., $k = 11$) values may result in underfitting.
- *Distance Metric:* the measure calculates the similarity or distance between instances. Depending on the application, different distance metrics, such as Euclidean or Manhattan distance, can significantly influence classification performances.

2. Random Forest

Random forest is an ensemble learning algorithm that combines multiple decision trees to make predictions [21]. Each decision tree in the random forest is built using a randomly selected subset of the training data and features. During prediction, the random forest aggregates the predictions of all the trees to make a final decision. By combining the predictions of multiple trees, random forest improves generalization and reduces the risk of overfitting, which is a common problem of traditional decision trees. This approach makes it more robust to noise and outliers in the data. The hyperparameters that we explore for this ML algorithm are noted below.

- *Number of Trees:* the number of decision trees T to include in the random forest. Increasing the number of trees can improve performance but also increases computational complexity.
- *Percentage of Data for Each Tree:* the training data percentage for each decision tree. Using smaller percentages introduces additional randomness and diversity in the forest, helping to mitigate overfitting.
- *Maximum Tree Depth:* the maximum depth allowed for each decision tree. Setting a maximum depth helps prevent overfitting by limiting the complexity of individual trees.
- *Number of Features per Tree (m):* the number of features considered at each split point in a decision tree. Smaller values increase randomness and diversity among trees, while larger values may produce more similar trees.

3. Support Vector Machine

SVM is a classification algorithm that finds an optimal hyperplane in the feature space to separate different classes [22]. It utilizes different kernel functions to transform the data into a

higher-dimensional space where a linear separation is possible. Then, a set of training instances closest to the decision boundary, known as support vectors, is used to find the hyperplane for the separation. For this investigation, we used the following hyperparameters.

- *Kernel Type:* the type of kernel function to use, such as linear, polynomial, or radial kernel. Each kernel has its impact on the decision boundary and model performance.
- *Kernel-Specific Hyperparameters:* each kernel type has its own set of hyperparameters. For example, the polynomial kernel has a degree parameter that controls the degree of the polynomial function, and the radial kernel has a gamma parameter that determines the kernel width. These hyperparameters influence the complexity and flexibility of the decision boundary.
- *Regularization Parameter* (C): it balances training error and margin size trade-offs. The margin separates the decision boundary and the nearest support vectors. A small C emphasizes a wider margin and allows more training misclassification, promoting generalization. A large C fits the training data more precisely but may lead to overfitting.

4. Artificial Neural Network

An artificial neural network is a machine learning model designed to mimic the functioning of the human brain to perform tasks such as pattern recognition and decision-making [20]. The key components of an ANN include input nodes, hidden layers, and output nodes. During training, the network learns to adjust the weights associated with connections between neurons to optimize its performance on a given task. The hyperparameters for tuning an ANN are crucial in achieving better generalization and preventing overfitting. Here are some essential hyperparameters to consider.

- *Activation Function:* this function acts as a decision maker for each neuron, determining whether the incoming information is significant enough to trigger the neuron's activation. Common activation functions include logistic, hyperbolic tangent (tanh), and rectified linear unit (relu).
- *Hidden Layers:* these are the middle layers between input and output, processing complex patterns using weighted connections and activation functions.
- *Number of Neurons*: the number of neurons in each layer, especially in the hidden layers, influences the network's ability to capture intricate patterns. More neurons may enhance the network's capacity to learn, but it also increases computational complexity.

Funding. H2020 LEIT Information and Communication Technologies B5G-OPEN (101016663).

Acknowledgment. We thank Professor Paolo Serena for his contribution to the generation of the simulation database.

REFERENCES

1. P. Layec, A. Morea, F. Vacondio, *et al.*, "Elastic optical networks: the global evolution to software configurable optical networks," Bell Labs Tech. J. **18**, 133–151 (2013).
2. M. Grieves, "Virtually intelligent product systems: digital and physical twins," in *Complex Systems Engineering: Theory and Practice* (2019), pp. 175–200.
3. D. Wang, Z. Zhang, M. Zhang, *et al.*, "The role of digital twin in optical communication: fault management, hardware configuration, and transmission simulation," IEEE Commun. Mag. **59**(1), 133–139 (2021).
4. T. Sasai, M. Nakamura, T. Kobayashi, *et al.*, "Revealing Raman-amplified power profile and Raman gain spectra with digital backpropagation," in *Optical Fiber Communication Conference (OFC)* (2021).
5. A. May, F. Boitier, E. Awwad, *et al.*, "Receiver-based experimental estimation of power losses in optical networks," IEEE Photon. Technol. Lett. **33**, 1238–1241 (2021).
6. T. Tanimura, S. Yoshida, K. Tajima, *et al.*, "Fiber-longitudinal anomaly position identification over multi-span transmission link out of receiver-end signals," J. Lightwave Technol. **38**, 2726–2733 (2020).
7. J. Müller, T. Fehenberger, S. K. Patri, *et al.*, "Estimating quality of transmission in a live production network using machine learning," in *Optical Fiber Communication Conference* (2021).
8. F. J. V. Caballero, D. J. Ives, C. Laperle, *et al.*, "Machine learning based linear and nonlinear noise estimation," J. Opt. Commun. Netw. **10**, D42–D51 (2018).
9. A. S. Kashi, Q. Zhuge, J. C. Cartledge, *et al.*, "Nonlinear signal-to-noise ratio estimation in coherent optical fiber transmission systems using artificial neural networks," J. Lightwave Technol. **36**, 5424–5431 (2018).
10. Z. Wang, A. Yang, P. Guo, *et al.*, "OSNR and nonlinear noise power estimation for optical fiber communication systems using LSTM based deep learning technique," Opt. Express **26**, 21346–21357 (2018).
11. A. Salehiomran and Z. Jiang, "Fast BER distribution and neural networks for joint monitoring of linear and nonlinear noise-to-signal ratios," in *Optical Fiber Communication Conference (OFC)* (2020).
12. P. Serena, C. Lasagni, and A. Bononi, "The enhanced Gaussian noise model extended to polarization-dependent loss," J. Lightwave Technol. **38**, 5685–5694 (2020).
13. M. Lonardi, P. Serena, P. Ramantanis, *et al.*, "Kerr nonlinearity dominance diagnostic for polarization-dependent loss impaired optical transmissions," in *European Conference on Optical Communication (ECOC)* (2021).
14. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., Springer Series in Statistics (Springer, 2009).
15. O. Vassilieva, S. Oda, T. Hoshida, *et al.*, "Experimental investigation of the statistics of the interplay between nonlinear and PDL effects in polarization multiplexed systems," in *Optical Fiber Communication Conference/National Fiber Optic Engineers Conference* (2013).
16. A. Bononi, N. Rossi, and P. Serena, "On the nonlinear threshold versus distance in long-haul highly-dispersive coherent systems," Opt. Express **20**, B204–B216 (2012).
17. K. Knight, *Mathematical Statistics* (CRC Press, 1999).
18. C. D. Cantrell, *Modern Mathematical Methods for Physicists and Engineers* (Cambridge University, 2000).
19. N. Rossi, S. Musetti, P. Ramantanis, *et al.*, "The impact of Kerr nonlinearity on the SNR variability induced by polarization-dependent loss," J. Lightwave Technol. **37**, 5048–5055 (2019).
20. G. James, D. Witten, T. Hastie, *et al.*, *An Introduction to Statistical Learning* (Springer, 2013).
21. L. Breiman, "Random forests," Mach. Learn. **45**, 5–32 (2001).
22. N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods* (Cambridge University, 2001).