Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Artificial Intelligence (37^{th} cycle)

Learning-based Methods for Fault Detection in Fiber-Optic Networks

Monitoring under Domain Shifts and Limited Supervision

Antonino Maria Rizzo

* * * * *

Supervisors
Prof. Giacomo Boracchi, Supervisor
Prof. Luca Magri, Co-Supervisor

Politecnico di Torino
2025

I hereby declare that the contents and organization of this dissertation constitute my own original work and do not compromise in any way the rights of third parties, including those relating to the security of personal data.

. . . . . .

Antonino Maria Rizzo

Turin, 2025

Acknowledgements

I want to thank *Giacomo* and *LucaM* for their mentorship and for introducing me to the art of transferring knowledge. Thank you for entrusting me with responsibilities that contribute to my professional and personal growth.

I am deeply grateful to the *Cisco* team, especially *Pietro* and *Enrico*, whose support has been indispensable throughout this journey. This work began with your passion for *Artificial Intelligence*, and this opportunity exists because of you. A special thank you to *Claudio* for his invaluable explanations of optical phenomena and his endless patience. His insights have been essential, particularly when writing Chapter 2, where he helped bridge the gaps in my optical knowledge. I also want to give a big thanks to *Gabriele* for all the time we spent debugging, troubleshooting, and figuring things out together. It was challenging yet fun.

To my colleagues – *Loris*, *Filippo*, *Edoardo*, *Giuseppe*, *Elena*, *Olmo*, *Roberto*, *Riccardo*, *Michele*, *LucaF*, and *Diego* – thank you for being there. Our travels, post-work aperitifs, beach volleyball games, ping-pong matches, and weekends at Loris' place have been a powerful source of energy throughout these years.

To my lifelong friends – *Federico*, *VitoP*, *Bilel*, *Domenico*, *PeppeS*, *Stefano*, *PeppeD*, and *VitoI* – thank you for your unwavering support, no matter the distance. Though we are scattered across Europe, our bond remains strong. Your presence in my life remains a source of endless strength and motivation.

Finally, I am deeply grateful to my parents for always believing in me, encouraging my curiosity and perseverance, and teaching me that every challenge is an opportunity for growth. To *Eleonora* – thank you for your patience, understanding, and firm support. Your belief in me, even in the most challenging moments, has been my anchor. Thank you for always lifting me up and reminding me to celebrate the small victories along the way. Your love and encouragement have been my greatest source of strength, and I am grateful to have you by my side.

This journey would not have been the same without all of you.

Summary

Fiber-optic networks form the backbone of modern telecommunication systems, enabling high-speed data transmission worldwide. Effective monitoring of these networks is essential to identify faults and prevent data loss. However, two major challenges arise in this context: *domain shifts* and *limited supervision*.

Domain shifts occur when the training data differs from those encountered during inference. These discrepancies limit the portability of monitoring solutions, often requiring models to be retrained for each new device. Additionally, limited supervision stems from the high costs and specialized expertise needed to collect annotated datasets, particularly when tailored to specific devices.

Our contribution to improving monitoring in fiber-optic networks is threefold. We first focus on monitoring *Performance Measures* (PMs), which reflect the transmission status at network nodes. Due to the non-linear effects of the optical components, PMs may exhibit significant statistical changes, even when the system is in the *in-control* state, i.e., operate under normal conditions. The *in-control* state is defined solely by a training set, and any deviation not represented in this set is considered an *out-of-control* condition, which must be detected as it indicates a potential fault in the system. Our proposed framework, termed *Hierarchical Change and Anomaly Detection* (HiCAD), effectively distinguishes between these states without relying on device-specific expert rules. It identifies candidate changes, extracts descriptors to capture their underlying patterns, and evaluates whether the detected changes remain within the *in-control* state. If not, an alarm is triggered to indicate a potential issue.

Our second contribution focuses on detecting faults in fiber links, which is critical in large networks and developing regions where breakages occur more frequently. *Optical Time-Domain Reflectometer* (OTDR) devices capture light propagation through the fiber and generate traces that reveal issues, such as faulty connectors or broken fibers, in the form of optical *events*. While manual inspection of these traces is possible, automated methods are preferred for efficiency. Therefore, we developed a 1D *Event-Detection Network* that learns

end-to-end to detect events of known classes. Moreover, we exploit an open-set classifier to recognize *unknown events*, i.e., categories of events not included in the training set. Additionally, we address the issue of domain shifts among traces from different OTDR devices. In particular, we designed an *Event-Detection System* that enables models trained on one device to be effectively applied to others by decoupling *localization* from *classification*, using signal processing techniques for event localization and a data-driven model for event classification.

Our third contribution focuses on monitoring optical spectra acquired by *Optical Channel Monitoring* (OCM) modules. Spectra provide a comprehensive view of the signal's power distribution across wavelengths, helping to identify faults that appear in the form of anomalies, i.e., channels that fail to reach their target power levels. During transmission, the non-linear effects of amplifiers and fiber spans introduce distortions that significantly affect the transmitted spectra, making it challenging to detect anomalies. Since distortions occur consistently across the entire spectrum, our method leverages the similarity of the spectrum trends to robustly estimate the expected power of the channels for each frequency location. Specifically, we identify candidate channels using robust fitting techniques and promote trend similarity through our proposed *joint optimization* procedure. Anomalies are detected as deviations from the trend of the channels.

Our advanced monitoring methods effectively learn from limited data and generalize across domains, adapting to various acquisition devices.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In modern communication systems, fiber-optic networks are the primary technology for high-speed data transmission and form the backbone of global telecommunication infrastructures. These networks are essential for connecting data centers, transmitting large volumes of internet traffic over long distances, and providing last-mile connections to homes and apartments. Fiber-optic cables traverse various and often impervious environments, from city streets to remote areas and undersea routes. Hence, they are vulnerable to external vibrations caused by heavy traffic, construction activities, earthquakes, tsunamis, or other natural disasters that can harm the network, resulting in poor transmission quality, higher latency, signal degradation, or even complete communication outages.

Advanced monitoring techniques are required to maintain a healthy infrastructure, especially for companies like *Cisco* that manage large-scale fiber-optic networks. However, there are two major challenges: *domain shifts* and *limited supervision*. Domain shifts occur because different devices employ various sensors and operate under different conditions. Indeed, when a model is trained on data from one device, it might perform poorly on other devices due to mismatches between the training data (source domain) and the data encountered during inference (target domain). This limits the portability of monitoring solutions, as models built for a specific type of device cannot generalize to data from other devices without extensive reconfiguration or retraining. Addressing domain shifts is crucial for maintaining the effectiveness of monitoring techniques across diverse network environments.

The limited supervision problem arises from the time-consuming task of collecting annotated datasets. This process requires expert knowledge to accurately label the observed optical phenomena, and it is further exacerbated when acquiring device-specific datasets. Moreover, gathering data under faulty conditions

might require extensive laboratory simulations or field acquisitions, which may be restricted by privacy concerns. Such data can expose sensitive information about a customer's internal systems. Also, faults are inherently rare, complicating the collection of diverse and representative data. Without enough faulty samples, machine learning models may become biased toward normal conditions, reducing their effectiveness in real-world fault detection.

Our works address these challenges by designing advanced methods that effectively learn from limited data availability and generalize to different domains. Our approach allows us to handle various acquisition systems and operating conditions, which is essential in scenarios where experts struggle to define precise criteria for identifying faults. Our contributions are further detailed below.

1.1 Research Contributions

Our contribution is threefold, addressing the development of specialized monitoring methods tailored to the distinctive characteristics of fiber-optic signals, effectively addressing their specific challenges.

1.1.1 Performance Measures Monitoring

First, we address the monitoring of Performance Measures (PMs) in Routed Optical Networks (RON) [25], which result in data streams describing the status of the transmission. These PMs represent an essential source of information for identifying poorly performing fiber links and anticipating failures or ongoing deteriorations that may affect specific components and disrupt normal operations. Timely fault detection is particularly challenging because the statistical properties of the PMs evolve over time, resulting in significant statistical changes even when the system is operating in the *in-control* state, i.e., under normal conditions. This evolution reflects the inherent dynamics of the transmission system, a phenomenon referred to by domain experts as the system's "breath". The *in-control* state is determined based solely on a training set, meaning that any variation not captured within this set is deemed an *out-of-control* condition. Identifying such deviations is crucial, as they may indicate potential faults in the system. Fig. 1.1 illustrates an example with two PMs. The green and red regions indicate the in-control and out-of-control states, respectively.

To achieve effective fault detection, we employ Change Detection (CD) tests. Traditional CD techniques [5, 81, 35] typically assume that samples are independent and identically distributed (i.i.d.) or adhere to a temporal dynamic that

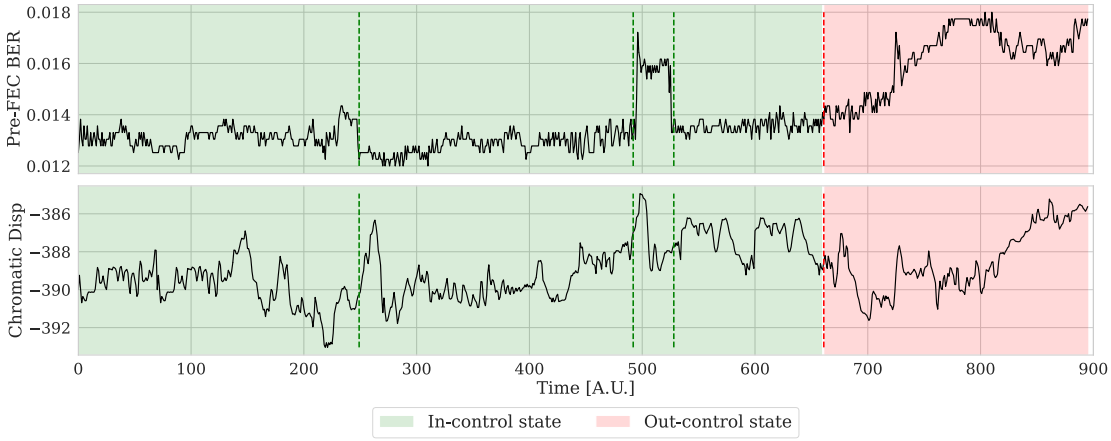


Figure 1.1: Examples of in-control and out-of-control states are represented by the green and red regions, respectively. The two PMs, Pre-FEC BER and Chromatic Dispersion (described in Chapter 2), show various statistical changes over time. However, not all of these changes indicate faults or degradation of the fiber link. The green dashed lines represent changes associated with the in-control state, while the red dashed line marks a degradation that causes the system to enter an out-of-control state.

can be reliably estimated. However, this assumption is too restrictive for our setting, causing CD tests to raise false alarms when changes associated with the in-control state occur. We developed a CD method that promptly detects only the changes marking the transition from the *in-control* state to an *out-of-control* state. Addressing this problem is crucial for reducing false alarms, allowing us to avoid unnecessary actions that could result in significant losses. Our novel framework, *Hierarchical Change and Anomaly Detection* (HiCAD), shown in Fig. 1.2, is structured into three layers, each serving a distinct role:

- The *Detection Layer* (DL) continuously monitors the data stream to detect candidate distribution changes. The DL preferably employs non-parametric CD tests, which can detect statistical changes without any assumptions on the underlying data distribution.
- The *Embedding Layer* (EL) identifies the windows surrounding the detected candidate changes and computes descriptors that capture their patterns. In particular, descriptors are standardized to address domain shifts between data streams collected under varying operating conditions.

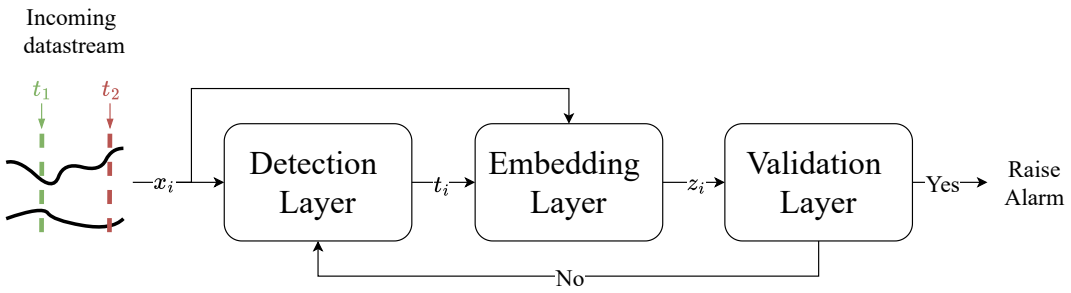


Figure 1.2: Our method consists of three main components: the Detection Layer (DL), the Embedding Layer (EL), and the Validation Layer (VL). The DL continuously monitors the data stream using a Change Detection Test to identify statistical changes as they occur. When a change is detected at time t_i, it triggers the subsequent layers. The EL encodes the relevant features of the change into a descriptor z_i. The VL evaluates whether the detected change marks the transition to an out-of-control state.

- The *Validation Layer* (VL) assesses whether the detected changes are associated with the in-control state or indicate the transition to an out-of-control state. This layer employs an anomaly detection model trained on descriptors of changes collected in the in-control state.

The key insight is to capture the normal behavior of the monitored system by learning from statistical changes. Any deviations from this model indicate a change to an out-of-control state. During training, the DL filters out likely stationary samples, as they do not provide helpful information for identifying changes. Therefore, the EL projects only in-control state changes into the descriptor space. This enables the VL to focus solely on these changes and learn a model that describes them. It is important to note that descriptors must capture common patterns of change, thereby forming high-density regions in the descriptor space. This allows descriptors of changes that lead to an out-of-control state to fall into low-density regions, making them easier to detect. Experiments conducted on real-world and synthetic data demonstrated that HiCAD effectively overcomes CD tests, including hierarchical ones, addressing false alarms while maintaining a low detection delay. The proposed approach has been submitted for a patent in collaboration with *Cisco*.

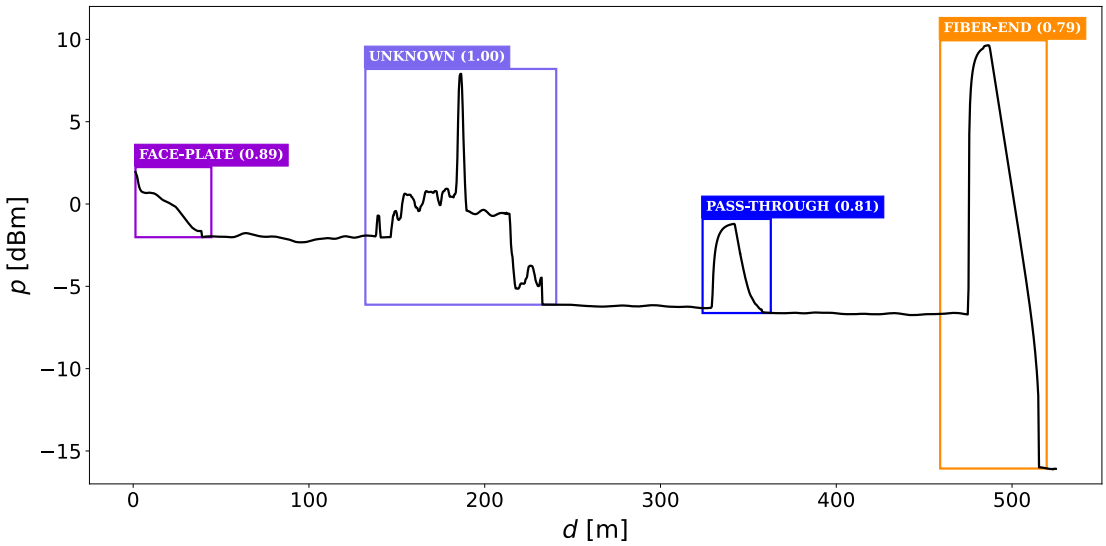


Figure 1.3: An OTDR trace that is 500 meters long and contains three known events, namely a Face-Plate, which appears as a downward slope on the left, a Pass-Through, which appears as a bump in the middle of the trace, and a Fiber-End on the right in which we have a sudden decay of the signal and then noise. There is also an unknown event associated with an attenuator installed at the beginning of the link. This qualitative example illustrates that our algorithm can detect unknown events in addition to known ones.

1.1.2 Fiber-Optic Event Monitoring

The Optical Time-Domain Reflectometer (OTDR) [62] is a widely used instrument for testing optical fiber links, revealing spurious optical reflections and power losses along the fiber. This device sends optical pulses through the fiber and measures the light scattered or reflected back, generating an *OTDR trace* (see Fig. 1.3). OTDR traces reveal distinctive patterns called *events*, each linked to a specific type of fault. These events are caused by Rayleigh scattering and Fresnel reflections, bends, or knots, indicating potential issues along the fiber link. By analyzing OTDR traces, faults can be identified and precisely located, enabling operators to prepare the appropriate tools for quick resolution.

Existing monitoring solutions implemented on embedded devices use simplistic detection rules that only classify events into broad categories such as *reflection*, *loss*, and *dead zones*. However, these broad categories are not sufficiently informative. Therefore, we aim for a more fine-grained event classification. Furthermore, we focus on detecting rare faults that emerge over time. Since these

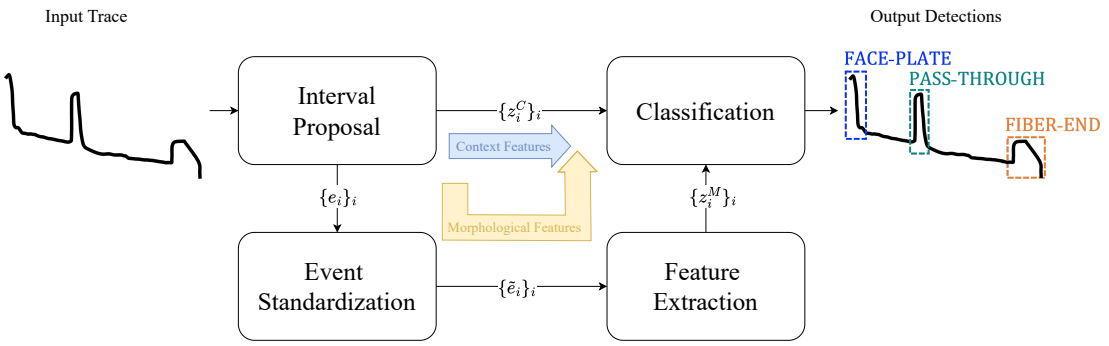


Figure 1.4: Overview of our Event-Detection System. The proposed *OTDR Event-Detection System* decouples localization from classification, enabling event standardization. In the Interval Proposal stage, we localize events and extract context features, such as event power and position. These events are then standardized in size and power. A feature extractor is trained to capture morphological characteristics combined with the context features to train the classifier.

faults fall outside predefined categories, they are often misclassified or completely missed [28], requiring ad-hoc procedures to detect them effectively.

We addressed OTDR monitoring by developing an *OTDR Event-Detection Network* learned end-to-end to detect an arbitrary number of known events corresponding to the labeled categories in the training set. Moreover, our method detects unknown events with a preprocessing stage that detects local peaks and classifies fixed-size windows around them using an open-set classifier based on *OpenMax* [6]. This method estimates the probability of an unknown event by analyzing the distribution of activation vectors from the training data. Specifically, it adjusts the components of the input's activation vector based on their likelihood to the fitted distribution, resulting in a modified vector termed the *recalibrated activation vector*. The difference between the original and recalibrated vectors reflects the network's uncertainty in its prediction and the probability that the input event is unknown. This approach not only sets a threshold for the network's confidence but also directly analyzes the distribution of the activation vectors. Therefore, it improves generalization, increases robustness against unknown events, and enables effective monitoring in dynamic real-world settings.

While our Event-Detection Network can be trained end-to-end, domain shifts limit its generalization ability across different OTDR devices. Models trained on data from one OTDR device struggle to adapt to others with varying sensor resolutions and operating modes, preventing cross-device deployment without additional training data from the target domain. To address these domain shifts,

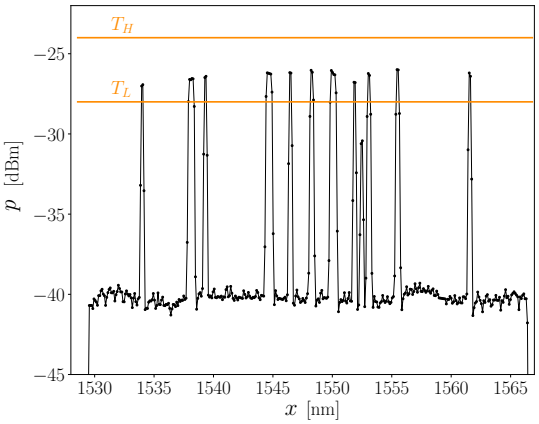
we proposed a novel *OTDR Event-Detection System*, depicted in Fig. 1.4. Our approach introduces the *Interval Proposal* stage, which combines signal processing techniques with robust fitting methods to identify candidate events and their *context features*, which capture domain-specific details such as power intensity and event position within the OTDR trace. For event classification, we standardize the events to obtain features that are independent of the input domain. These features, referred to as *morphological* features, focus on the shape of the events while disregarding context-specific details. Morphological features are combined with the context features generated by the IPA and fed into a classifier to determine the event type. By integrating both context and morphological features, our method ensures accurate predictions and can be effectively deployed across different devices.

Experiments on real-world traces from two types of OTDR devices by *Cisco* demonstrate that our method achieves a mAP@0.5 of 75.33% on traces of the kind of device used for training, while the 1D Faster R-CNN reaches a mAP@0.5 of 69.27%. When applied to traces from new types of devices, our method still performs well, attaining a mAP@0.5 of 64.93% without fine-tuning. These results indicate a 50% improvement in mAP over traditional methods, validating our framework's adaptability and effectiveness across different device types. Our method is deployed on both the *NCS1001* [23] and *NCS1010* [21] devices.

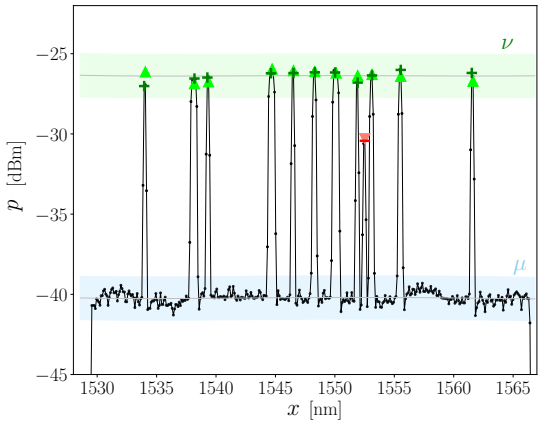
1.1.3 Optical Spectra Monitoring

A popular tool to remotely monitor optical spectra transmitted through fiber is the Optical Channel Monitoring (OCM) module [22], which can reveal issues such as signal attenuation and wavelength shifts.

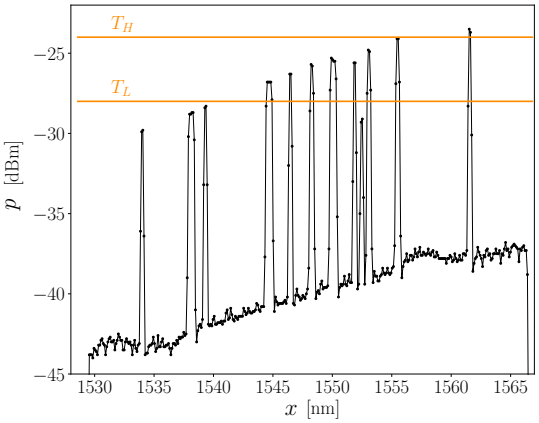
Optical spectra, depicted in Fig 1.5, provide a comprehensive view of the signal's power distribution across various wavelengths. A spectrum consists of channels that appear as bumps with equal target power levels (see Fig. 1.5a). Channels typically pass through multiple amplifiers and spans, leading to several distortions, including: *i)* tilt and ripple from EDFAs and distributed Raman amplifiers, *ii)* wavelength-dependent loss in the transmission fiber, *iii)* stimulated Raman Scattering (SRS) effects between channels, *iv)* spectrum distortion from add/drop sections. As shown in Fig. 1.5, distortions significantly affect the transmitted spectra, making it challenging to detect *anomalies*, which appear as uneven peaks, marked by downward red triangles in Fig. 1.5b. Spectrum anomalies indicate fiber faults. Therefore, it is essential to implement procedures for detecting them and triggering corrective actions. Without such procedures, autonomous systems that enhance transmission quality (e.g., optical amplifiers)



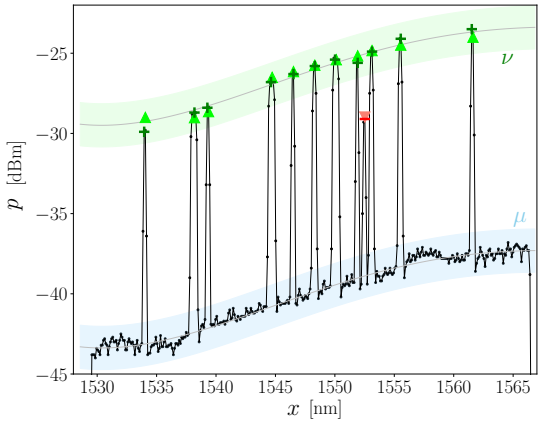
(a) Two-threshold approach tested over an equalized spectrum.



(b) Joint Optimization tested over an equalized spectrum.



(c) Two-threshold approach tested over a distorted spectrum.



(d) Joint Optimization tested over a distorted spectrum.

Figure 1.5: In a all the channels appear as peaks in the spectrum having the same power level (equalized), whereas in c the spectrum becomes distorted and the channels are no more equalized. While in a, the thresholds T_H and T_L (colored orange) can successfully separate channels from anomalies, in c, the use of thresholds becomes ineffective. Figures b and d depict the channel and ASE trends, ν and μ, estimated by our proposed method.

could introduce erroneous adjustments, potentially leading to cascading effects.

Traditional approaches for anomaly detection in optical spectra involve setting two power thresholds, T_L and T_H (orange lines in Fig. 1.5a), to identify anomalies as peaks outside these boundaries. However, these methods are only effective on well-equalized spectra, which are rare due to amplifier fluctuations and tilts that distort the spectrum, causing both channels and ASE to follow a

trend, denoted as ν and μ in Fig. 1.5d. When the spectrum is transmitted, these trends are no longer horizontal, and the two-threshold approach fails, as illustrated in Fig. 1.5a and Fig. 1.5c.

To detect anomalies even with heavy distortions and tilts, we propose estimating the channel trend and identifying anomalies as deviations from it. To capture the principal trends in the spectrum, we employ robust fitting techniques, such as RanSaC [33]. Since distortions occur consistently across the entire spectrum, we expect the trends to be similar. Therefore, our proposed joint optimization procedure promotes trend similarity. Specifically, we first identify the candidate channel and noise samples and compute two initial guesses of ν and μ using robust estimators [33]. We then refine the initial guesses of ν and μ by solving a non-linear optimization problem using the Levenberg-Marquardt [67] algorithm. We jointly optimize the two trends ν and μ by minimizing the following loss:

$$\arg\min_{\mu,\nu} \sum_{p\in\mathcal{C}} \text{err}(p,\nu)^2 + \sum_{p\in\mathcal{N}} \text{err}(p,\mu)^2 + \frac{\lambda}{d}\sum_{j=1}^{d}(n_j - m_j)^2 \tag{1.1}$$

where $\text{err}(p,\nu)$ represents the distance (over the vertical axis) of the trend ν to samples $p \in \mathcal{C}$, and similarly $\text{err}(p,\mu)$ represents the distance between samples $p \in \mathcal{N}$ and the trend μ. The first two terms of Eq. (1.1) represent the data fidelity of the trends to the samples, whereas the third term enforces the similarity of the two trends, allowing ν and μ to be close to each other, except a translation along the vertical axis that is encoded in their zero-degree terms n_0 and m_0. The parameter $\frac{\lambda}{d}$ balances the data fidelity and the similarity terms. In particular, the division by d allows setting the same value of λ for different order d of the trends. Note that the values n_j e m_j are the trend coefficients optimized at each optimization iteration. Finally, anomalies are recognized as peaks $p \in \mathcal{C}$ that fall outside a predefined inlier threshold ε, which defines a green band around the channel trend. Peaks falling outside this band are classified as anomalies.

Experiments on real-world spectra gathered from *Cisco* have shown that our method is accurate and capable of detecting anomalies even in the presence of severe distortions and tilts. Quantitative analysis on a dataset of synthetically generated spectra, modeling the main distortions that occur in the fiber, confirms the advantages of our approach, as we achieve more than 98% accuracy. The proposed approach has been patented and deployed to the NCS1001 [22] system.

1.2 Published Work

This thesis includes excerpts from the following articles and patents, which I authored and which present the most significant outcomes of my research:

- **A. M. Rizzo**, M. O. Nogara Notarianni, P. Invernizzi, C. Crognale, C. Alippi, L. Magri, G. Boracchi. "*HiCAD: Data-driven, Hierarchical Change Detection in Non-stationary Data Streams*". IEEE Transactions on Neural Networks and Learning Systems (TNNLS). Under Submission.
- P. Invernizzi, C. Crognale, R. Manzotti, G. Martinelli, **A. M. Rizzo**, G. Boracchi, C. Alippi, M. O. Nogara Notarianni, L. Magri, S. Binetti. "*Semi-Supervised Hierarchical Monitoring Of Performance Measures In Routed Optical Networks*". Submitted Patent.
- **A. M. Rizzo**, L. Magri, P. Invernizzi, E. Sozio, S. Piciaccia, A. Tanzi, S. Binetti, C. Alippi, G. Boracchi. "*Anomaly Detection in Optical Spectra via Joint Optimization*". 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2023.
- **A. M. Rizzo**, L. Magri, P. Invernizzi, E. Sozio, G. Aquaro, S. Binetti, C. Alippi, G. Boracchi. "*Event Detection in Optical Signals via Domain Adaptation*". 2023 31st European Signal Processing Conference (EUSIPCO), 2023.
- **A. M. Rizzo**, L. Magri, D. Rutigliano, P. Invernizzi, E. Sozio, C. Alippi, S. Binetti, G. Boracchi. "*Known and unknown event detection in OTDR traces by deep learning networks*". Neural Computing and Applications (NCA), 2022.
- C. Alippi, G. Boracchi, **A. M. Rizzo**, L. Magri, E. Sozio, P. Acharjee, P. Invernizzi, S. Piciaccia. "*Automatic Trend Identification and Anomaly Detection in Optical Channel Monitoring Spectrum*". Accepted Patent, 2022.

The following publication was developed in collaboration with colleagues from the *National PhD program in Artificial Intelligence* and presents an automatic tool for monitoring 1D acoustic signals:

- J. Melchiorre, L. D'Amato, F. Agostini, **A. M. Rizzo**, "*Acoustic emission onset time detection for structural monitoring with U-Net neural network architecture*", Developments in the Built Environment, 2024.

As this topic does not fit the topic of this thesis, it will not be further discussed in this manuscript.

1.3 Outline

This thesis is structured as follows:

- Chapter 2 provides the necessary background on fiber-optic networks, focusing on Dense Wavelength Division Multiplexing, Routed Optical Networks, faults, and the optical devices employed for monitoring, such as the Optical Time-Domain Reflectometers (OTDR), and Optical Channel Monitoring modules.
- Chapter 3 addresses the monitoring of Performance Measures (PMs) in RON. We formalize the problem, review relevant PM monitoring and Change Detection literature, and describe our methodology. The chapter concludes with a discussion of the experimental results.
- Chapter 4 details the detection of both known and unknown events in OTDR traces across different device types. We define the problem and review related works on object detection networks, deep learning for time series, and open-set recognition. We then discuss our proposed solutions for detecting unknown events and addressing domain shifts.
- Chapter 5 focuses on anomaly detection in optical spectra. We describe our channel and ASE trend estimation approach through our joint optimization procedure. The chapter also presents a comprehensive experimental evaluation, comparing various methods and discussing the results.
- Finally, Chapter 6 provides concluding remarks, summarizing our research contributions and suggesting directions for future work.

Chapter 2

Background on Fiber-Optic Networks

Effective monitoring of optical networks is critical for maintaining high performance and reliability in modern optical transmission systems. To maintain the network's integrity and enable efficient traffic management, typical monitoring devices are *i) transceiver performance monitors* [24], *ii) Optical Time-Domain Reflectometers* (OTDR) [62], and *iii) Optical Channel Monitoring* (OCM) modules [22]. Specifically, the performance monitor embedded in optical transceivers collects the main parameters that characterize transmission performance at network nodes. The OTDR analyzes the backscattered and reflected light to reveal breaks or signal loss. Meanwhile, the OCM monitors the individual optical channels, showing their power distribution along the optical spectrum.

Below, we provide a basic overview of optical fibers, transmission systems, and networks, describing common faults and impairments that can affect them. Finally, we discuss the monitoring devices mainly used to detect these issues.

2.1 Optical Fibers

A modern fiber-optic system can transmit data over long distances, often thousands of kilometers, properly exploiting the propagation of the optical field through fiber-optic cables [1, 27]. The basic architecture of an optical transmission system (see Fig. 2.1) is made by *i)* an *optical transmitter* that converts the electrical signals into optical ones using light sources (e.g., laser diodes), *ii)* an optical fiber cable as transmission medium, and *iii)* an *optical receiver* that captures the optical signal at the receiving end and retrieves the original data.

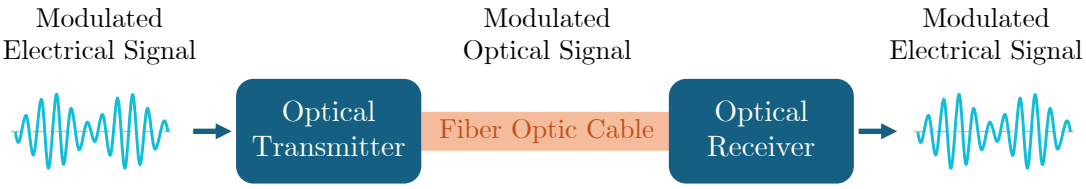


Figure 2.1: Basic architecture of an optical transmission system.

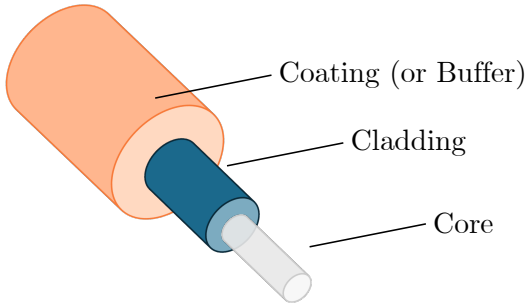


Figure 2.2: Structure of a fiber-optic cable.

A single optical fiber is composed of three main parts: the *core*, the *cladding*, and the protective *coating* or *buffer* (see Fig. 2.2). The central core is a cylindrical rod of dielectric material, with a diameter ranging from 8 μm to 63, μm through which the light propagates. Surrounding the core is the cladding, which is also made of dielectric material but has a lower refractive index than the core. This arrangement forms an optical wave-guide that confines light within the core through total internal reflection at the core-cladding boundary. When light reaches the core of an optical fiber at the critical angle specified by Snell's law [63], it undergoes internal reflection, continuously bouncing off the core's internal surfaces with the fiber acting as an electromagnetic surface wave-guide. Light traveling through the optical fiber achieves data transmission at 99.7% of the speed of light in a vacuum [75]. The outer coating protects the cladding from physical damage (e.g., bending or abrasion) or environmental factors (e.g., temperature fluctuations). As a result, the light remains confined within the core, enabling transmission over long distances with minimal signal loss.

Optical fibers are categorized into single-mode and multi-mode types based on their core diameter [63]. Single-mode fibers, with a small core diameter of around 8-10 μm, confine light to a single propagation mode, reducing signal attenuation and distortion. This makes them well-suited for long-distance communications, such as transcontinental data transmission. In contrast, multi-mode fibers, which

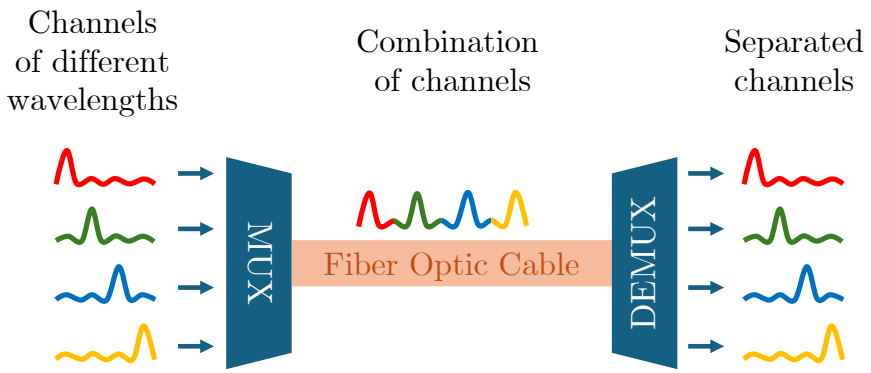


Figure 2.3: In DWDM transmission, optical channels of different wavelengths are combined by a multiplexer (MUX) and transmitted simultaneously through the fiber-optic cable. At the receiving end, a demultiplexer (DEMUX) separates these channels and directs them to their intended destinations.

have a larger core diameter of approximately 50-62.5 μm, are less effective over long distances due to modal dispersion. As a result, they are commonly used in short-range communication systems, such as local area networks (LANs).

Fiber-optic networks have completely transformed data transmission, offering unmatched speed, efficiency, and security. They accommodate large bandwidth demands while transmitting data over long distances with minimal loss, making them essential for modern communication infrastructure.

2.2 Dense Wavelength Division Multiplexing

Dense Wavelength Division Multiplexing (DWDM) is an optical transmission technology that sends multiple data channels simultaneously over a single optical fiber, enhancing the overall data throughput. This is achieved by assigning each channel to a specific wavelength of light, effectively aggregating several signals into one fiber. The process, illustrated in Fig. 2.3, begins with a multiplexer (MUX) that combines multiple optical signals – each at a different central wavelength[1] – into a single beam of light. A demultiplexer (DEMUX) separates these signals at the receiving end and directs them to the appropriate destinations.

[1]Depending on system requirements, either the channel's wavelength λ or frequency f can be used. The well-known relationship $\lambda f = c$ [34] describes the conversion between them, where c is the speed of light in a vacuum. This manuscript uses the terms "wavelength" and "frequency" interchangeably for convenience.

Moreover, optical amplifiers are deployed at various points in the network to maintain the strength of the light signals as they travel over long distances. These amplifiers, such as Erbium-Doped Fiber Amplifiers (EDFAs), boost the signal without converting it back to an electrical format, ensuring that the data can continue traveling large distances with minimal signal loss. Amplifiers are typically categorized based on their position in the link as follows: *i)* booster amplifiers (BSTs), placed after the MUX to enhance input optical power; *ii)* in-line amplifiers (ILAs), distributed along the link to periodically compensate for optical losses; and *iii)* pre-amplifiers (PRE), located at the end of the link to increase optical channel power before demultiplexing.

In DWDM optical systems, when the receiver detects errored bits, preprocessing algorithms correct these errors and enhance performance. The most commonly used error correction algorithm is the *Forward Error Correction* (FEC) [1]. To evaluate the effectiveness of the FEC, we measure the *Bit Error Rate* (BER) [13] both before and after correction. The pre-FEC BER indicates the number of erroneous bits before any correction is applied, while the post-FEC BER measures the number of residual erroneous bits remaining after correction. It is important to note that the effectiveness of the FEC depends on the input pre-FEC BER. When the pre-FEC BER exceeds a critical value, known as the *pre-FEC BER threshold* or *pre-FEC BER at FEC*, the FEC may not be able to correct all bits, and error-free transmission cannot be guaranteed. Therefore, the post-FEC BER reflects the actual number of errors detected after transmission.

To transmit large volumes of data, we use *coherent transmission* [13, 44, 1], which processes the optical field's phase and amplitude. This technology optimizes the use of optical fibers and enables high-capacity data transfer[2].

2.3 Routed Optical Networks

Routed Optical Networks (RONs) [24, 42] are modern optical transmission networks that address the limitations of traditional network architectures, which are often inefficient due to the numerous line cards for traffic management and overlapping control and management systems. These problems contribute to high costs, poor resource utilization, and hindered automation, resulting in longer service delivery times [26].

RONs can be either static or dynamic. In static routing, data paths through the

[2]In this manuscript, we consider only optical systems based on coherent transmission.

network are predefined and remain fixed. While this approach is more straightforward to manage, it can be less efficient, especially when network conditions change. On the other hand, dynamic routing allows the network to adjust the paths in real-time based on factors such as traffic load, congestion, or failures. Dynamic routing enables the network to optimize performance and ensure reliability by adapting to current conditions.

RONs enable fast, scalable, and reliable long-distance data transmission by efficiently routing data through optical fibers, dynamically adjusting to network conditions, and reducing costs, power consumption, and hardware requirements with high-density routers and high-capacity optical devices.

2.4 Faults and Impairments

Fiber impairments can degrade signal quality to varying degrees. In severe cases, they can disrupt the connection entirely, stopping the propagation of the optical signal. In less severe instances, impairments may introduce bit errors without completely interrupting transmission. The main impairments that might occur in optical networks, such as RONs, are:

- **Amplified Spontaneous Emission (ASE):** Noise contribution introduced by amplifiers. It adds up to the transmitted signal, hence limiting the system performance at the receiver side [13].

- **Filtering:** As introduced in Sec. 2.2, we can simultaneously transmit multiple optical channels exploiting multiplexers (MUX) and demultiplexers (DEMUX). These devices introduce a filtering impairment on the channels, cutting out frequency components from the channels *unilaterally* or *bilaterally*.

  An example of the former typically can result from channel frequency *detuning*, which refers to a misalignment between the central frequency of the channel and that of the fiber. The latter can arise when the channel has a spectral width that exceeds the filter size, causing spectral components to extend beyond the filter's edges [1].

- **Chromatic Dispersion:** Optical fibers transmit at different speeds the field components at different frequencies, leading to a progressive distortion of the signal propagating through the optical fiber [27].

- **Linear XTalk:** When multiple channels are transmitted simultaneously, each channel can interfere with the others, causing a partial overlap of their spectral width in the receiver [1].

- **Polarization Mode Dispersion (PMD):** Minor deviations from perfect cylindrical symmetry in the fiber give rise to the phenomenon of stochastic "birefringence", affecting the orthogonally polarized components of the field. When the input pulse propagates through the fiber, it broadens as the two polarization components disperse along the fiber due to their different velocities of propagation induced by the stochastic birefringence [1].
- **Nonlinear Interference (NLI):** The Nonlinear Interference (NLI) [74, 12] is due to the fiber nonlinearities [1]. When the overall fiber length is large (thousands of km) and the DWDM comb is made of several coherent channels, it can be considered with a good approximation as an additive Gaussian white noise into the receiver bandwidth, similar to the ASE noise.
- **Loss:** Loss of power caused by the attenuation that the optical signal experiences during the propagation in the fiber link.

Notice that the level of impairment determines the severity of the fault in the system. Under normal conditions, ASE and detuning are moderate, resulting in a system working properly. However, as the level of ASE and detuning increases, we enter a critical condition where the system continues to operate but with suboptimal performance. This behavior can be quantified by an increase in the pre-FEC BER at the receiver. When the pre-FEC BER threshold is exceeded, errors in the data occur.

2.5 Optical Monitoring

This section describes the most commonly used monitoring techniques experts rely on to characterize transmission, optimize DWDM optical link performance, and detect faults, impairments, and anomalies.

2.5.1 Performance Monitoring

Transceivers built into routers control data transmission and reception and gather performance measures related to the network's status at the node. This data enables optical experts to evaluate the health of the fiber and identify potential issues that could indicate fiber damage or poor link performance. The key parameters typically used by optical experts for performance monitoring are:

- **Bit Error Rate:** As discussed in Sec. 2.2, the Bit Error Rate (BER) [13] measures the number of bit errors per unit of time. BER can be measured before

or after applying the FEC algorithm, resulting in *pre-FEC* and *post-FEC BER*, respectively. Some systems report the number of uncorrected blocks (*UCB*), which reflects groups of errored bits. When the UCB value is greater than zero ($UCB > 0$), it indicates that the FEC could not correct all bit errors, potentially leading to system failure.

- **Q-Factor:** It is customary to use a quality parameter derived from the *pre-FEC BER*, B, referred to as the *Q-Factor* [1] which is defined as:

$$Q = 2 \cdot \left(\text{erfc}^{-1}(2 \cdot B)\right)^2 \tag{2.1}$$

where the erfc is the *complementary error function* [1]. In practice, the *Q-Factor* is commonly expressed in logarithmic units:

$$Q_{dB} = 10 \cdot \log_{10}(Q)\,. \tag{2.2}$$

- **Optical Signal-to-Noise Ratio:** The Optical Signal-to-Noise Ratio (OSNR) [13] represents the ratio of the optical signal power to the noise power within a specified optical bandwidth (e.g., 0.1 nm) [13]. The higher the OSNR, the better the signal quality and the lower the probability of bit errors in the transmission. The OSNR is typically measured in decibels (dB).

- **Transmitter/Receiver Input Power:** Coherent optical interfaces typically monitor *i)* the optical channel power at the transmitter side, *ii)* the total optical power received at the receiver side, which includes the power carried by the DWD channel comb and by ASE, and *iii)* the received optical channel power, which refers specifically to the power of the individual channels.

- **Polarization-Dependent Loss:** Polarization-Dependent Loss (PDL) [13] occurs when light passes through optical devices such as amplifiers or filters, resulting in time-dependent fluctuations in the output optical power. This power loss, which depends on the wavelength, can significantly impair signal quality and degrade overall system performance.

2.5.2 Optical Reflections and Power Losses Monitoring

Optical Time-Domain Reflectometers (OTDR) systems operate like one-dimensional radars inside optical fibers. They can accurately locate faults along a fiber link (e.g., spurious optical reflections and power losses) with a spatial resolution of up to a few centimeters. The underpinning principle is illustrated in

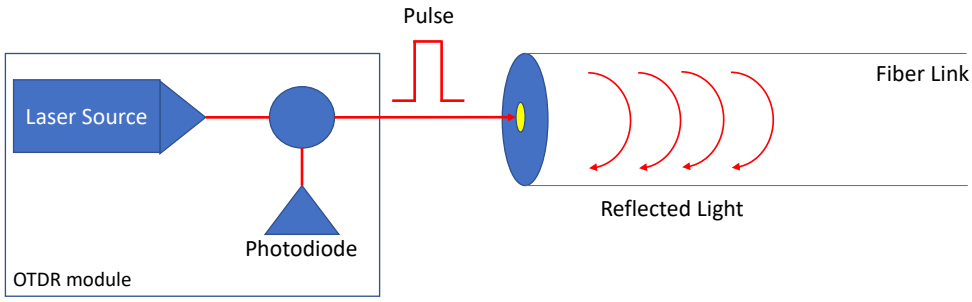


Figure 2.4: The typical OTDR module includes a laser that sends a pulse to the fiber link and a photodiode that gathers the backscattered and reflected signal.

Fig. 2.4: a short laser pulse is injected at one end of the fiber link, and a photodiode at the same location measures the backscattered and reflected signal. The OTDR trace accumulates thousands of measurements and represents the optical power (expressed in dBm) as a function of the distance along the fiber (expressed in km). An example of the resulting trace is reported in Fig. 2.5.

The analysis of OTDR traces allows the identification of a large number of failures, termed *events*, that are primarily due to the following optical conditions:

- **Rayleigh Back Scattering:** It represents the natural reflection of the fiber, which is due to impurities and inhomogeneous structures resulting from the fabrication process. Rayleigh Back Scattering contributes to the attenuation of optical power as a function of the fiber distance, resulting in a linear decay trend of OTDR traces, usually expressed in dB/km.

- **Fresnel Reflection:** It originates when the laser pulse hits a physical device such as a connector, mechanical splice, bulk attenuator, or fiber break. Upon impact, part of the light pulse is reflected toward the source, typically a photodiode, resulting in a local increase in the reflected light.

- **Concentrated Losses:** These are due to attenuators, splices, or splitters that reduce the optical power returned to the OTDR instrument, and the OTDR trace attenuates very abruptly. Finally, when the fiber contains two nearby physical objects, the former might introduce reflection or scattering that masks the latter. Regions where the optical measurements suffer from such masking effect are named *dead zones*.

These optical conditions, illustrated in Fig. 2.5, generate optical signatures, also termed *events*, that optical experts, or specific software, analyze to diagnose impairments on the fiber links.

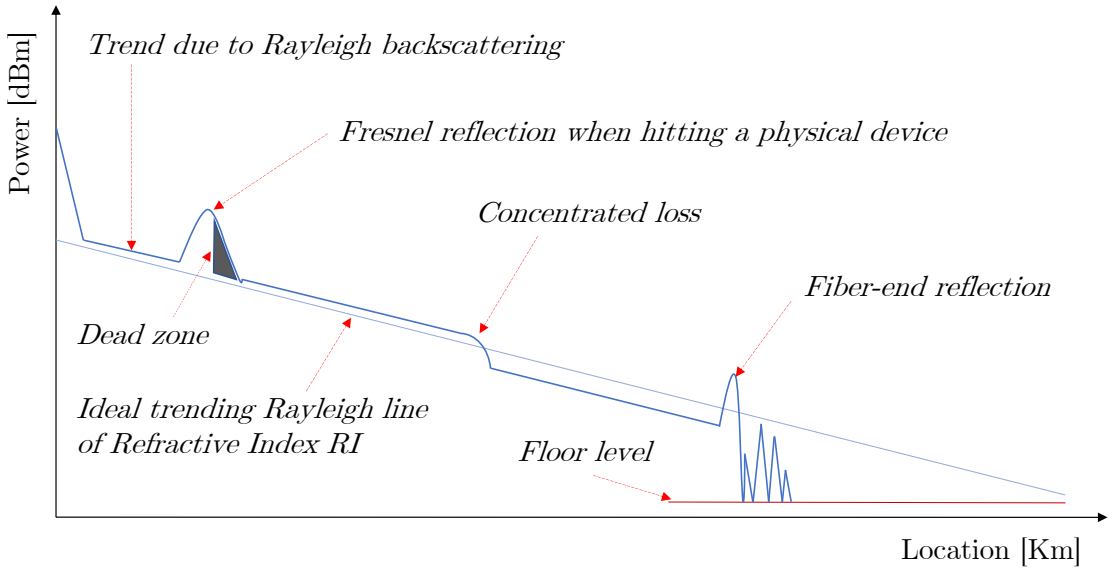


Figure 2.5: Illustration of an OTDR trace with multiple events. The text annotations describe the root causes of these events.

2.5.3 Optical Channel Monitoring

Recent photonic technological advances have enabled accurate power level monitoring across multiple optical channels in DWDM links. This capability is increasingly exploited for tasks such as channel power equalization, amplifier channel gain tuning, and anomaly detection. This feature, known as Optical Channel Monitoring (OCM), is now commonly embedded into various optical devices within fiber-optic networks. For example, in recent years, OCM modules have been incorporated into optical amplifiers at terminal nodes (e.g., PRE and BST) or along the fiber link (e.g., ILA). At a high level, OCM modules simultaneously sample the power of the transmitted DWDM spectrum at the amplifier's input or output with a specified optical resolution (e.g., 12.5 GHz), depending on the technology of the embedded device. Fig. 2.6 illustrates a typical end-to-end connection between two terminal nodes, A and B, where OCMs are embedded into amplifiers. Terminal node A transmits a spectrum composed of four channels with similar power levels. However, an apparent anomaly is observed in the third channel, where the power deviates from the expected range, defined by the upper threshold (T_H) and lower threshold (T_L).

After transmission through the link, the received spectrum at terminal node B appears distorted, making anomaly detection more challenging. Due to these distortions, the predefined thresholds no longer accurately reflect the expected

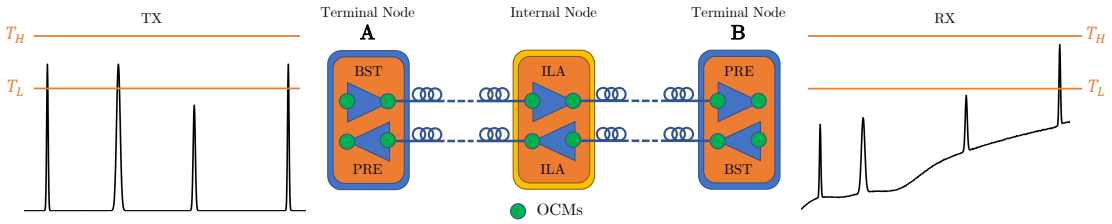


Figure 2.6: Scheme of an end-to-end connection between two terminal nodes. Green dots represent the positions where OCM scans can be performed. On the left (TX side), we depict an example spectrum suitable for transmission, whereas the received spectrum, distorted in the transmission, is displayed on the right (RX side).

channel power levels. New procedures must account for the fiber link distortions to detect anomalies correctly. OCM modules can be deployed at terminal nodes and along the fiber link, significantly improving network reliability and enabling the transmission of channels over longer distances.

Chapter 3

Monitoring Performance Measures in Routed Optical Networks

Routed Optical Networks (RON) are designed to efficiently and scalably manage traffic routing over long fiber links [25]. Performance Measures (PMs), representing transmission status, are commonly collected to monitor system behavior and identify potential faults (described in Sec. 2.4). However, analyzing these PMs is challenging because their statistical properties evolve over time, leading to changes even when the system remains in an *in-control* state, i.e., operating under normal conditions. Such allowed changes may be confused with those that indicate an *out-of-control* state, suggesting potential faults. In practice, the in-control state is defined by a training dataset that captures the system's natural evolution, a phenomenon experts refer to as the system's "breath". Any deviation beyond what is represented in this dataset is considered a change to an out-of-control state. An example is shown in Fig. 3.1, where three PMs exhibit changes within the in-control state (green dashed lines) and a change to the out-of-control state (red dashed line), which precedes a degradation in the system's operating conditions. We aim to detect only changes that mark the transition from the *in-control* to an *out-of-control* state, allowing the system to adapt to different devices without relying on fixed thresholds or expert-defined rules. Early fault detection is essential for maintaining network performance, as it identifies potential issues before they occur. This proactive approach allows network operators to reroute traffic in advance, minimizing the impact of potential degradations, preventing data loss, and ensuring consistent quality of service.

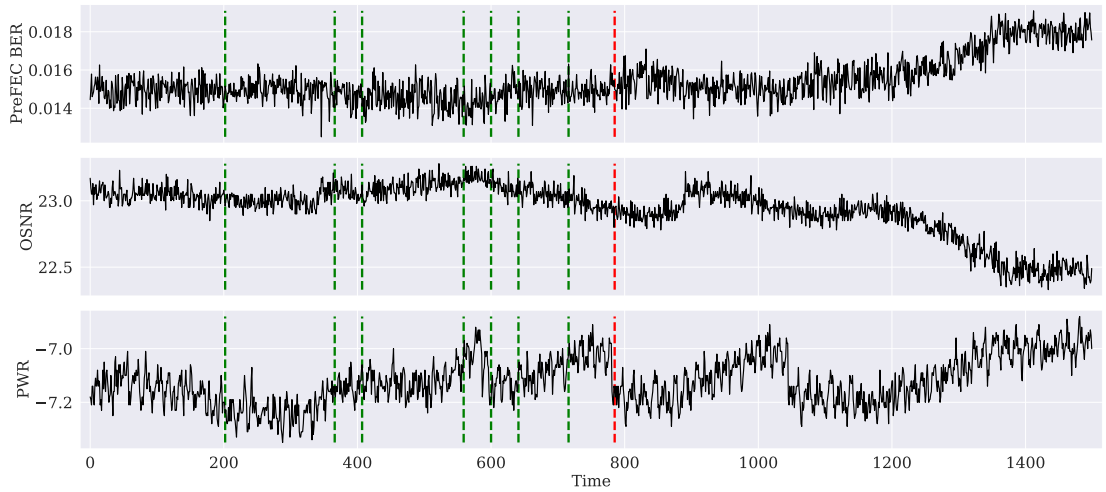


Figure 3.1: This visualization depicts three Performance Measures (PMs) collected in *Cisco* laboratories: *PreFEC BER*, *OSNR*, and *PWR*. These metrics illustrate the system's behavior over time. The green dashed lines represent changes within the in-control state, while the red dashed line marks the transition to an out-of-control state, after which significant degradations are observed in all three metrics.

Existing monitoring techniques rely on transmission-specific thresholds to analyze individual PMs, providing only partial insights into the overall system status. Change Detection Tests (CDTs) can identify statistical changes in the data distribution. However, these tests typically assume that the input data is independent and identically distributed (i.i.d.), a condition that does not hold in our scenario, where data samples exhibit complex temporal dynamics. Furthermore, while CDTs detect all statistical changes, they do not distinguish between those occurring within the in-control state and those indicating a transition to an out-of-control state, resulting in many false alarms. To address this problem, we designed a framework, *Hierarchical Change and Anomaly Detection* (HiCAD), to detect changes to out-of-control states, filtering out irrelevant alarms based on *Cisco*'s experience. HiCAD consists of three main components. First, the *Detection Layer* (DL) identifies potential changes in the data-generating process using a CDT. When multiple PMs must be analyzed simultaneously, we employ a multivariate CDT to fully grasp the system status. When a change is detected, it activates the subsequent layers for further analysis. Second, the *Embedding Layer* (EL) selects PM windows around the detected changes and creates descriptors that capture their patterns. Finally, the *Validation Layer* (VL) assesses

whether the candidate changes mark the transition to an out-of-control state by providing their descriptor to an anomaly detector. This detector is trained on a dataset of descriptors derived from changes observed in the in-control state.

Overall, our novel hierarchical framework enhances monitoring reliability by effectively detecting statistical changes in the system while minimizing false alarms. This allows domain experts to anticipate fiber deteriorations and maintenance needs, optimize network operations, and enable the early detection of faults before they escalate. Additionally, our method is applicable across a wide range of devices and operating conditions, as it operates independently of transmission-specific thresholds.

3.1 Problem Formulation

We operate in an online setting. The monitored system generates a continuous data stream, and each new sample must be processed as soon as it is collected. The data stream can be either univariate or multivariate. PM samples are typically scalar in fiber-optic network monitoring, but statistical changes may occur across multiple PMs. Therefore, it is crucial to analyze multiple PMs jointly, resulting in a truly multivariate monitoring scenario. Our approach makes minimal assumptions about the data-generating process, which we model as a stochastic process $\mathcal{P} : \mathbb{R}^+ \to \mathbb{R}^d$ that produces a d-dimensional sequence $X(t) = \{x_i(t)\}_{i=1}^d$ for each time t. This sequence is not necessarily composed of independent and identically distributed (i.i.d.) realizations of a random variable.

Under regular system operation – free from faults, degradation, or abnormal behavior – we define the in-control process as $\mathcal{P}_0$. We assume that samples are drawn from one of its in-control modalities, denoted as $\{\mathcal{P}_{0,m}\}_{m=1}^M$. These modalities may alternate over time and generate data streams that consist of either i.i.d. samples or samples with temporal dependencies. The number of in-control modalities, M, and their specific characteristics remain unknown. The in-control process is represented solely by a training set, $\text{TR} = \{X_i\}_{i=1}^N$, consisting of N sequences of realizations from $\mathcal{P}_0$. We assume that this training set implicitly captures the system's normal behavior, encompassing samples from all M modalities. We denote the out-of-control process as $\mathcal{P}_1$, representing deviations from regular system functioning. This process is also unknown and its realizations are not included in the training set. We define the change-point τ as the time instant when the realizations of $\mathcal{P}$ transition from the in-control state

to an out-of-control state, i.e.:

$$x(t) \sim \begin{cases} \mathcal{P}_{0,m} & \text{if } t < \tau \\ \mathcal{P}_1 & \text{if } t \geq \tau. \end{cases} \tag{3.1}$$

Since $\mathcal{P}_0$ and $\mathcal{P}_1$ are in general non-stationary processes, we consider either abrupt, incremental, or gradual changes [60] and assume the transition to be part of the post-change process $\mathcal{P}_1$. We aim to promptly detect when the system evolves towards an unseen *out-of-control* condition, namely $\mathcal{P}_1$. The proposed solution has to be applied to data streams acquired under varying operating conditions without prior knowledge of the specific system configurations or the need for retraining, thereby addressing the *domain shift*.

3.2 Related Literature

3.2.1 Machine Learning in Fiber-Optic Monitoring

The growing demand for bandwidth has led to major advancements in optical networks, making them increasingly sophisticated, configurable, and adaptable. As these networks become more complex, efficient performance monitoring and optimization have become crucial. Traditional network monitoring methods primarily depend on predefined thresholds, with fault identification being identified by domain experts. However, recent research highlights the potential of Machine Learning in optical networks, attracting considerable interest in the field [94].

One such approach is the Bit Error Rate (BER) Anomaly Detection (BANDO) algorithm [93], which focuses on identifying significant BER changes within optical connections. To monitor the BER evolution over time, BANDO establishes an outer boundary to anticipate BER threshold violations and detect sudden faults. Additionally, two inner boundaries are set to trigger the re-estimation of the outer boundary whenever the measured BER exceeds these limits. The inner boundaries are defined as $\mu \pm k \cdot \sigma$, where μ and σ are the mean and standard deviation computed from the last n BER measures, and k is a scaling factor that varies for each boundary. The BANDO algorithm handles three distinct scenarios. When the measured BER exceeds one of the inner boundaries, BANDO re-estimates the outer boundary and sends a notification to the network controller with a low severity level. When the measured BER exceeds the outer boundary, BANDO issues a warning notification to the network controller. Finally, when the measured BER exceeds the BER threshold, BANDO sends a notification to the network controller with a major severity level, indicating a fault.

Similarly, algorithms in [8] track the State of Polarization (SOP) to monitor mechanical stress on optical fibers. These algorithms trigger an alert when SOP speed surpasses a set threshold, helping to predict fiber breaks. In contrast, the Testing optIcal Switching at connection SetUp timE (TISSUE) algorithm [92] is designed to detect and localize failures by comparing estimated BER values against theoretical BER values. When the slope difference between the two exceeds a predefined threshold, TISSUE anticipates potential failures. While these techniques focus on single PMs (univariate monitoring) and threshold-based detection, our method captures the complex interactions between multiple transmission parameters simultaneously, offering a more comprehensive approach to performance monitoring.

Further advances in fault detection are seen in methods such as those proposed in [18], which use Optical Spectrum Analyzers (OSAs) to measure optical power at different wavelengths. A density-based clustering approach leverages similarities among data samples to classify them into clusters and identify outliers. Samples collected in normal conditions tend to cluster in high-density regions, while post-fault samples, being rare, fall into low-density regions and are flagged as outliers. In another approach, Kruse et al. [48] employ optical spectra from OSAs as input to a Variational AutoEncoder (VAE) trained for fault detection. Specifically, the VAE is optimized during training to minimize reconstruction errors on non-faulty data. When a fault occurs, the reconstruction error increases, allowing for fault detection through a threshold-based system. These methods operate on spectral data, which is inherently more complex than the PMs we focus on. Notably, all the discussed methods monitor a single channel at a time, whereas our approach enables simultaneous monitoring of PMs across one or multiple channels, offering a more comprehensive solution.

3.2.2 Hierarchical Change Detection

Change Detection (CD) involves identifying shifts in the input data distribution. While existing CD literature primarily assumes the input samples to be independent and identically distributed (i.i.d.), real-world data frequently exhibit temporal dynamics and changes associated with the in-control state. In such cases, detrending and decorrelation techniques can be used, but they may still produce correlated samples or completely hide statistical changes, hindering the effectiveness of the monitoring. The *Hierarchical Change Detection Test* (HCDT) [4] is one of the earliest hierarchical frameworks consisting of a detection and a validation layer. The detection layer employs CDTs with low detection delay and minimal computational cost. Once a change is detected, the validation layer verifies the

change in data distribution to reduce false positives and improve detection accuracy. HCDT assumes that the input data is i.i.d. thus relying on a statistical test in the validation layer. In contrast, we address a more complex scenario where the input data is not i.i.d., making traditional statistical tests unsuitable for validating changes in distribution. To handle this, our method learns a model of the system's normal behavior directly from the data.

Interestingly, hierarchical approaches received very little attention in the Change Detection literature while several techniques [60, 102] have adapted this framework for *concept drift detection*, which involves detecting changes in the relationship between input data and labels to prevent Machine Learning models from becoming outdated. Concept drift detection methods usually depend on labeled data, which can be challenging to obtain or unavailable. Our approach, however, requires only a set of data streams collected under normal operating conditions. It employs a learning-based validation layer rather than traditional Change Detection tests. Furthermore, while many techniques exist for concept drift detection, none of them uses a data-driven validation layer.

In the Anomaly Detection literature, [73] applies an anomaly detector directly to each incoming sample, followed by a hierarchical framework that identifies changes and updates the trained detector to prevent it from becoming outdated. This technique detects changes on a sample-by-sample basis using rule-based heuristics. In contrast, rather than analyzing individual samples, our method performs Anomaly Detection by considering the regions preceding the changes identified by a Change Detection test. As a result, our method effectively detects changes marking the transition to an out-of-control state, leading to more accurate and robust fault detection.

3.3 Methodology

Our novel framework, *Hierarchical Change and Anomaly Detection* (HiCAD), is designed to be modular, allowing for the selection of the most suitable modules at each layer to address the specific characteristics of the problem at hand. This section outlines the methodology behind HiCAD, while Sec. 3.4 presents candidate methods that can be integrated at each layer. The resulting hierarchical structure is depicted in Fig 3.2.

HiCAD consists of three layers: *i)* the *Detection Layer* (DL), which identifies candidate changes; *ii)* the *Embedding Layer* (EL), which extracts descriptors capturing the patterns of the changes; and finally, *iii)* the *Validation Layer* (VL)

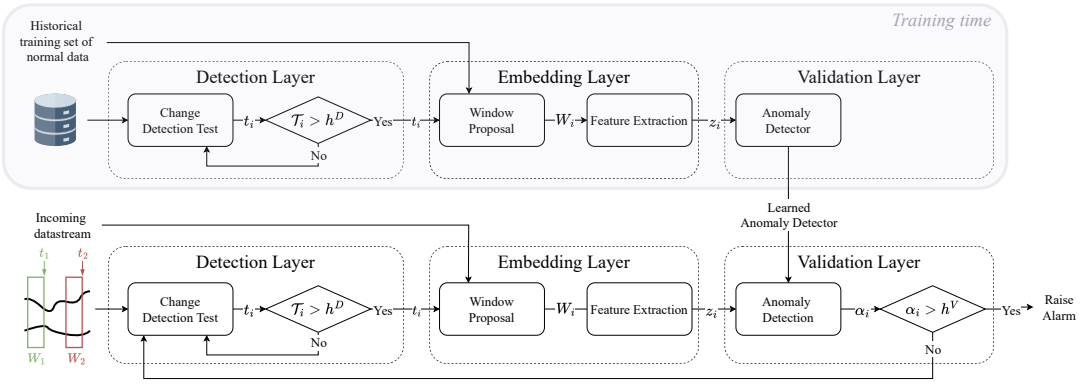


Figure 3.2: Our method consists of three main components: the Detection Layer (DL), the Embedding Layer (EL), and the Validation Layer (VL). The DL continuously monitors the data stream using a Change Detection Test to identify statistical changes as they occur. When a change is detected at time t_i, it triggers the subsequent layers. The EL selects a window W_i around the change-point t_i, extracts features, and encodes them into a descriptor z_i. The VL computes the anomaly score α_i for z_i to evaluate whether the detected change is part of the in-control process. As illustrated in the top row, the anomaly detector is trained using only examples of changes within the in-control process.

evaluates the descriptors and raises alarms only for changes that result in an out-of-control state. The complete algorithm is detailed in Alg. 1. Below, we provide a comprehensive description of its components.

3.3.1 Detection Layer

The Detection Layer (DL) consists of a Change Detection Test (CDT) that continuously monitors the input data stream by applying a hypothesis test to detect statistical changes in the data-generating process. For each incoming observation $X(t_i)$, the DL computes a statistic $\mathcal{T}_i$ (see Alg. 1, line 2). When $\mathcal{T}_i$ exceeds the threshold h^D, a candidate change is detected at time t_i, and the DL triggers the subsequent layers (see Alg. 1, line 3). The threshold computation depends on the CDT, as detailed in Sec. 3.4.4.

To build the DL, we select nonparametric and sequential CDTs that can process multivariate data streams in real-time with no assumption about the underlying data distribution. Sec. 3.4.1 provides examples of suitable CDTs for our framework. Since the DL employs a CDT, it relies on the assumption that either

Algorithm 1 HiCAD

Input: Data stream $X(t)$, thresholds h^D, anomaly score threshold h^V
Output: Change-point t^*, if detected

while $X(t)$ is available **do**
  Compute statistic $\mathcal{T}_t$ using a CDT
  if $\mathcal{T}_t \geq h^D$ **then**
    Define window $W_t \in \mathbb{R}^{w \times d} \leftarrow \{X(t-w), \ldots, X(t)\}$
    Compute descriptor z_t from window W_t
    Compute anomaly score α_t
    if $\alpha_t \geq h^V$ **then**
      return t ▷ Raise the alarm
    end if
  end if
end while
return -1 ▷ No alarm

i) data are i.i.d. from an unknown random variable, or *ii)* data adhere to a temporal dynamic that can be estimated using a single model, allowing one to compute i.i.d. residuals from the sequence. Before the CDT, we can apply an optional pre-processing step to detrend the data stream when it exhibits significant temporal dynamics. The pre-processing step is described in Sec. 3.3.1.

Pre-processing

The pre-processing step is optional and is applied only when the data stream exhibits significant temporal dynamics. Since the CDTs employed at the DL rely on the i.i.d. assumption, they would repeatedly trigger the EL and VL due to temporal trends. To address this issue, the data streams can be detrended by fitting models that capture these patterns. Common approaches include *AutoRegressive Moving Average* (ARMA) models [38], *Kalman Filters* [46], and *Exponential Smoothing* [11]. Specifically, once the model is fitted, we compute the differences between the measured samples and the predictions, i.e., the residuals, which can be conveniently provided as input to the DL.

It is important to note that while shallow detrending may not sufficiently mitigate temporal dynamics, overly aggressive detrending can hide changes to the out-of-control state. Therefore, we must strike a balance between alleviating temporal dynamics and preserving the changes we aim to identify.

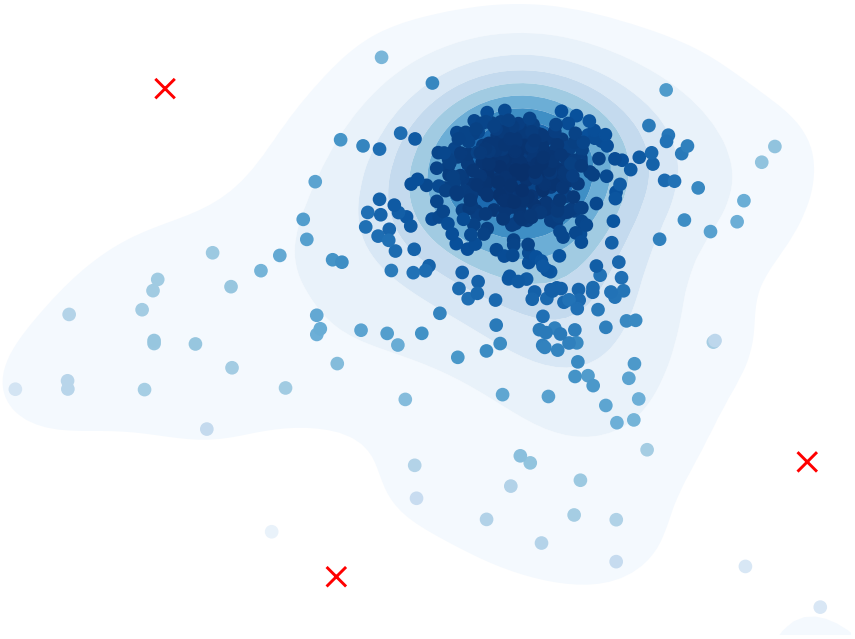


Figure 3.3: Example of descriptor space. The figure shows that descriptors of changes from the in-control state, depicted with blue dots, cluster together, while descriptors of changes to the out-of-control state, marked with red "X", fall in low-density regions. This space has been obtained using a dimensionality reduction technique, specifically Multidimensional Scaling [7], which projects descriptors from a 12-dimensional space into a 2D space.

3.3.2 Embedding Layer

The *Embedding Layer* (EL) operates in two stages to extract a descriptor z_i, which captures the characteristics of the candidate change at time t_i triggered by the DL. In particular, in the first stage, the *Window Proposal* module selects a window $W_i \in \mathbb{R}^{w \times d}$ surrounding t_i (Alg. 1, line 4), where w is the window length. In the second stage, the *Feature Extractor* computes the descriptor vector $z_i \in \mathbb{R}^k$ (see Alg. 1, line 5), which compactly represents the window W_i containing the potential change. To mitigate differences between data streams collected under varying operating conditions, the descriptors are standardized using a reference window W_0 captured at the beginning of the stream. This standardization is essential for ensuring consistency in the feature extraction process. Sec. 3.4.2 provides examples of suitable descriptors that can be used to build the EL.

3.3.3 Validation Layer

The *Validation Layer* (VL) evaluates whether the change at t_i indicates a transition to the out-of-control state. Fig. 3.3 presents a 2D representation of the embedded space, illustrating several projections of windows that contain potential changes. Descriptors computed by the EL from windows of the in-control process $\mathcal{P}_0$ – represented in the training set – tend to cluster together, reflecting the system's expected behavior. In contrast, descriptors derived from windows associated with faults deviate from these clusters. Therefore, we approach the validation step as an anomaly detection task.

Given a descriptor z_i, the VL computes its anomaly score α_i (see Alg. 1, line 6), which quantifies how much the corresponding window W_i deviates from the patterns observed in the training set. Higher scores indicate more significant deviations from the training sequence. A change-point is detected, and an alarm is raised if α_i exceeds a threshold h^V (see Alg. 1, line 7). The threshold computation procedure is described in Sec. 3.4.4.

The anomaly detector is trained with data streams collected during normal operating conditions. Thanks to the DL, the EL projects only windows containing statistical changes in the descriptor space. By filtering out stationary samples, the DL reduces clutter in the descriptor space and enhances the sensitivity of the VL. The training procedure for the VL is described in Sec. 3.3.4. Moreover, Sec. 3.4.3 provides examples of suitable anomaly detectors that can be used within our framework.

3.3.4 Training Procedure

The anomaly detector within the Validation Layer (VL) is trained on data collected when the system operates in normal conditions. This enables the detector to effectively learn the patterns characterizing the in-control process. By focusing exclusively on data collected from the in-control process, our framework is well-suited for real-world scenarios where collecting data from failure conditions is impractical, such as in the case of wide fiber-optic networks.

As illustrated in the top part of Fig. 3.2, the training procedure begins with the Detection Layer (DL). Given a set of training data streams, the DL identifies potential changes, which then trigger the Embedding Layer (EL) to select windows that capture the patterns of these changes. The EL extracts descriptors to create a compact representation, retaining only the essential information. We assume that the candidate changes are part of the in-control process and use them to train the feature extraction model in the EL and the anomaly detection model

in the VL. It is important to note that the anomaly detection models we use (see Sec. 3.4.3) are designed to be robust to a small fraction of faulty sequences in the training set without significantly compromising the performance.

3.4 Techniques for Implementing Layers

3.4.1 Change Detection Tests

The Detection Layer, in principle, includes any Change Detection Test (CDT). However, when monitoring multiple data streams in real-time, it must be *multivariate* and process samples as soon as they are collected. Furthermore, to address domain shifts, a suitable CDT has to be *nonparametric* to be independent of the underlying data distribution. The sections below outline sequential CDTs that can be integrated into our framework.

Normal Discrepancy

The Normal Discrepancy (ND) [50] is designed to detect dissimilarities across three sliding windows: the current window W_i, its left and right sub-windows, W_i^L and W_i^R, respectively. The ND is defined as:

$$d_N(W_i, W_i^L, W_i^R) = n \left(\ln|\Sigma_{W_i}| - \frac{1}{2}\ln|\Sigma_{W_i^L}| - \frac{1}{2}\ln|\Sigma_{W_i^R}| \right) \tag{3.2}$$

where n is the size of the window, and Σ_{W_i}, $\Sigma_{W_i^L}$, and $\Sigma_{W_i^R}$ are the covariance matrices of the current window W_i, the left sub-window W_i^L, and the right sub-window W_i^R, respectively.

The determinant of a covariance matrix, often referred to as the *generalized variance* [95], measures the dispersion of the observations in a given window. A large determinant indicates a more dispersed data cloud.

Before computing the ND for a window W_i, we standardize it relative to a reference window W_0. This is done as follows:

$$\tilde{W}_i = \frac{W_i - \mu_0}{\sigma_0} \tag{3.3}$$

where μ_0 and σ_0 represent the mean and standard deviation of W_0, computed separately for each data stream.

Scan-B

Scan-B [51] detects statistical changes in data streams using kernel-based, non-parametric methods. Specifically, it employs the Maximum Mean Discrepancy (MMD), a kernel-based statistic that quantifies differences between the distributions before and after a candidate change. MMD compares the means of data mapped into a Reproducing Kernel Hilbert Space (RKHS) to determine whether two sets of data come from the same distribution. In an online setting, Scan-B processes data samples using a sliding window that captures the most recent data points and compares them with reference blocks from past data. As new data arrives, the window shifts forward, and the MMD is recomputed. A change-point is detected when the standardized MMD exceeds a predefined threshold, which is set to control the false positive rate and ensure timely detection.

Semi-Parametric Log-likelihood

The Semi-Parametric Log-likelihood (SPLL) [49] describes the initial data distribution, denoted as ϕ_0, by fitting a Gaussian Mixture Model (GMM), $\hat{\phi}_0$, to the initial window W_0. New incoming data batches are then compared against batches from W_0 using a likelihood test. We exploit SPLL for dimensionality reduction of incoming samples by computing their log-likelihood with respect to the GMM $\hat{\phi}_0$. The resulting univariate sequence is then monitored with techniques such as Change-Point Model (CPM) or Cumulative Sum (CUSUM), described below.

SPLL Change-Point Model

The SPLL Change-Point Model (SPLL-CPM) [36] combines SPLL with the Change Point Model [81]. Since CPM is designed for univariate monitoring, we integrate it with SPLL for dimensionality reduction. The resulting univariate sequence is then monitored using CPM, leveraging the Lepage statistic (L) [81]. Although CPM offers theoretical guarantees for controlling false alarms with i.i.d. samples, our settings violate this assumption, limiting the model's ability to accurately represent the real data-generating processes.

SPLL Cumulative Sum

The Cumulative Sum (CUSUM) method [89] is a sequential monitoring technique designed to detect shifts in the process mean. The core idea is that, for a time series with a constant zero mean, the cumulative sum of its observations should follow a zero-mean Gaussian distribution, $\mathcal{N}(0, \sigma^2)$. Therefore, any significant

deviation of the cumulative sum from this distribution indicates the occurrence of a statistical change. Since CUSUM is designed for univariate signals, inspired by [36], we leverage the SPLL model and then apply CUSUM to the resulting univariate sequence.

3.4.2 Feature Extraction

The EL compactly represents the relevant features of the windows W_i, each containing a potential change proposed by the DL, as descriptor vectors z_i, which are then fed into the anomaly detector. Several feature extraction techniques can be employed to represent windows W_i. For instance, we compute the mean and variance of W_i to capture its macroscopic characteristics. Alternatively, we reduce dimensionality using Principal Components Analysis (PCA) [7] or an Autoencoder (AE) [7]. Another approach is decomposing the window into spectral components using the Discrete Cosine Transform (DCT). The sections below discuss these approaches in more detail.

Macroscopic statistics

To characterize the window W_i, we employ macroscopic statistics, specifically the mean μ and variance σ^2. We divide W_i into two sub-windows, left and right, to identify significant deviations between older and newer samples. For each window component, we compute its features as $(\mu_L, \sigma_L^2, \mu_R, \sigma_R^2)$, where the subscripts L and R refer to the left and right sub-windows, respectively. Finally, we construct the descriptor z_i by stacking the features of all the components into a single vector.

Principal Component Analysis

To reduce the dimensionality of the window W_i while extracting the most significant features, we employ Principal Component Analysis (PCA) [7]. In particular, we apply PCA to the training set to identify the k principal components that capture the most variance in the data. This process yields a transformation matrix $\tilde{\mathbf{P}}$. During testing, we use $\tilde{\mathbf{P}}$ to consistently transform the window W_i into the same feature space as the training samples. The resulting vector $z_i \in \mathbb{R}^k$ is then used as a feature descriptor. Notice that the principal components estimated from the training set are fixed at inference time. Formally, we first transform the 2D window $W_i \in \mathbb{R}^{w \times d}$ into a 1D representation of size $w \cdot d$. This transformation is achieved by spatially concatenating the d components of the input window. Given a training set $\mathbf{S} \in \mathbb{R}^{N \times wd}$, the mean is removed to yield

a zero-mean matrix $\bar{\mathbf{S}}$. The PCA decomposition leads to the expression $\bar{\mathbf{S}}\mathbf{P} = \mathbf{X}$, where $\mathbf{X} \in \mathbb{R}^{N \times wd}$ is the transformed data, and $\mathbf{P} \in \mathbb{R}^{wd \times wd}$ is the projection matrix. By selecting the top k principal components from $\mathbf{P}$, a reduced matrix $\tilde{\mathbf{P}} \in \mathbb{R}^{wd \times k}$ is obtained. The projection of the training set into this k-dimensional subspace, represented by $\bar{\mathbf{S}}\tilde{\mathbf{P}} = \tilde{\mathbf{X}} \in \mathbb{R}^{N \times k}$, serves as the descriptor set for the training sequence. Furthermore, the window can be reconstructed from these descriptors through the expression $\tilde{\mathbf{S}} = \tilde{\mathbf{X}}\tilde{\mathbf{P}}^{\top}$, resulting in a matrix $\tilde{\mathbf{S}} \in \mathbb{R}^{N \times wd}$, which can be used to monitor the reconstruction error (see 3.4.3).

Discrete Cosine Transform

To build descriptors, we represent input windows in the frequency domain using the Discrete Cosine Transform (DCT), which decomposes each window into a sum of harmonic cosine functions oscillating at different frequencies [2]. The DCT is more efficient than the Fourier Transform because it only captures the real part and is computationally faster. To identify the cosine waves that contribute the most to representing the training set, we apply the DCT to each window, average the resulting coefficients, and select the k largest ones. These coefficients correspond to the most significant frequency components and are expected to capture the key characteristics of the signal. When a new input window W_i is provided at test time, we apply the DCT and extract the same k frequency components identified during training, ensuring consistent feature extraction. The result is a compact descriptor, $z_i \in \mathbb{R}^k$. It is important to note that we compute the DCT for each PM independently and then concatenate the resulting descriptors. Formally, given a training set $\mathbf{S} \in \mathbb{R}^{(N \times w \times d)}$, we compute the DCT transform as $\mathbf{DS} = \mathbf{X}$, where $\mathbf{X} \in \mathbb{R}^{N \times w \times d}$ is the transformed data, and $\mathbf{D} \in \mathbb{R}^{w \times w}$ is the transformation matrix. We then average the N spectra and select the m components with the highest amplitude. We obtain the reduced transformation matrix $\tilde{\mathbf{D}} \in \mathbb{R}^{m \times w}$ by considering the m rows corresponding to these largest averaged components. The resulting matrix of descriptors $\tilde{\mathbf{X}} \in \mathbb{R}^{N \times m \times d}$ in the reduced frequency space is computed as $\tilde{\mathbf{D}}\mathbf{S} = \tilde{\mathbf{X}}$. Moreover, the window can be reconstructed from this reduced space using the expression $\tilde{\mathbf{S}} = \tilde{\mathbf{D}}^{\top}\tilde{\mathbf{X}}$, where $\tilde{\mathbf{S}} \in \mathbb{R}^{N \times w \times d}$ is the matrix of reconstructed signals. This can be used to monitor the reconstruction error (see 3.4.3).

Autoencoder

We can learn meaningful descriptors by applying dimensionality reduction through an Autoencoder (AE) [7]. In particular, since our training set only consists of data from normal operating conditions, the AE learns the features and

patterns that characterize the system's normal behavior.

After training, for each window W_i, we leverage the encoder to extract the latent representation z_i and use it as a descriptor. Since the training set only consists of realizations of the in-control process, the AE struggles to generate meaningful representations of out-of-control sequences that do not conform to the patterns learned Our AE has the following architecture. The encoder is composed of three convolutional layers, each followed by a GELU activation function and batch normalization. Each convolution reduces the sequence length by half. The decoder then reconstructs the data from the latent space back to the original space, using upsampling, additional convolutional layers, GELU activations, and batch normalization. The entire model consists of $13\,790$ parameters with an input window of size 96×2. Training is performed using the Adam optimizer, with the Huber loss being minimized to refine the reconstruction.

3.4.3 Anomaly Detection

To assess whether a potential change deviates from the in-control process, we explored both anomaly detection and reconstruction-based methods. In the first case, the VL consists of an anomaly detector that receives an input descriptor z_i and computes its anomaly score α_i. The higher the anomaly score, the larger the likelihood that the potential change, raised by the DL and represented by z_i, is not part of the in-control process. We tested two density-based anomaly detectors: the Kernel Density Estimation in Section 3.4.3 (distribution-based) and Isolation Forest in Section 3.4.3 (tree-based). In the latter case, we consider the EL to bypass the Feature Extraction step and output the observed window W_i, which is used to compute a reconstruction error through similar feature extraction techniques in the VL.

Kernel Density Estimation

We employ Kernel Density Estimation (KDE) [7] to estimate the unknown probability density function (PDF) of the descriptors extracted by the EL. KDE works by placing a kernel, such as a Gaussian function, at each descriptor vector and then summing or averaging these kernels to generate the overall PDF.

Unlike parametric methods, which assume a specific data distribution (e.g., Gaussian distribution), KDE is a nonparametric approach that adapts to the actual distribution of the data. This adaptability makes it particularly well-suited for analyzing irregularly distributed data in high-dimensional spaces.

Isolation Forest

Isolation Forest (IFOR) [58] detects anomalies by assessing how easily a data point can be isolated. The underlying principle is that anomalies are rare and significantly differ from the majority of the data, making them easier to isolate compared to normal samples. Specifically, IFOR counts the number of random splits needed to isolate a data point. The fewer the splits, the more likely the point is to be classified as an anomaly.

IFOR leverages *isolation trees*, which are built by recursively splitting the data based on randomly selected features and random split values. The final Isolation Forest is formed by combining multiple isolation trees. IFOR averages the path lengths across all the trees to compute the anomaly score for a data point.

It is worth noting that IFOR is a nonparametric method, meaning it does not assume any specific underlying distribution for the data.

Reconstruction-based

Monitoring reconstruction error is a common approach for Anomaly Detection in time series analysis [84]. The reconstruction error, defined as the difference between the observed window W_i and the predicted window $\hat{W}_i$, is typically measured using the Mean Absolute Error (MAE) and compared against a threshold h^V. If the error exceeds this threshold, the candidate change is considered part of the out-of-control process, indicating a significant deviation from normal behavior in the data. To obtain the predicted window $\hat{W}_i$, various techniques can be applied, such as Principal Component Analysis (PCA), Autoencoders (AE) for dimensionality reduction, or the Discrete Cosine Transform (DCT), which represents the window using its primary harmonic functions.

3.4.4 Threshold Computation

This section outlines the threshold computation methods for both the Detection Layer (DL) and Validation Layer (VL). To compute the thresholds for the DL, we employ different approaches based on whether the CDT is nonparametric or semi-parametric. In the latter case, the threshold computation procedure is data-independent. Since nonparametric CDTs do not rely on a specific data distribution, we rely on Monte Carlo simulations. In particular, we generate sample streams from the Gaussian distribution $\mathcal{N}(0, 1)$ and compute the statistic for each sequence. Since all samples are drawn from the same Gaussian distribution, we expect no changes, resulting in a low probability that the statistic

exceeds the threshold h^D. Consequently, we set h^D at the 99th percentile, averaging 100 000 simulations to determine the final value. This approach can be used with rank-based CDTs, such as the CPM with the Lepage statistic, and with any semi-parametric Gaussian test, such as the ND. Optical PMs can be effectively approximated by Gaussian distributions. As a result, thresholds estimated by sampling from $\mathcal{N}(0, 1)$ closely match the quantiles of the experimental distribution. For other semi-parametric CDTs, such as SPLL [49], we follow the procedure in the reference paper. Thresholds are estimated given an initial reference window, which is not a major limitation for PM monitoring. This is because the system runs for several days to ensure it functions properly before being used for production purposes.

To estimate thresholds for the VL, only sequences collected under normal conditions are required. After collecting a set of windows that triggered the DL, we extract the descriptors and compute the corresponding anomaly scores. We set the threshold h^V as the empirical 95th percentile of these anomaly scores.

3.5 Experiments

In this section, we evaluate HiCAD using both real-world and synthetic optical data. Our experiments demonstrate that HiCAD provides significant benefits in filtering out statistical changes that are part of the in-control process, overcoming traditional CDTs, and effectively reducing false alarms. We describe the considered datasets (Sec. 3.5.1) and figures of merit for the algorithms' performance assessment (Sec. 3.5.2), we discuss the experimental procedure and show the monitoring results (Sec. 3.5.3) and its limitations (Sec. 3.5.3).

3.5.1 Considered Datasets

Microsoft Wide Area Network

The Microsoft Wide Area Network (MS-WAN) dataset contains 14 months of optical PMs from Microsoft's backbone network in North America [65]. PMs are taken every 15 minutes and include the Q-factor [13], transmit power (dBm), and chromatic dispersion (ps/nm) [27].

The PMs contain interruptions due to outages that have been removed for confidentiality reasons. Since the data distribution may change suddenly before and after these interruptions, potentially leading to significant alterations, we remove them from the dataset. Specifically, we control the timestamps of the

PMs and retain only those sequences that have a minimum length of $1\,000$ samples. As a result, we obtain over $17\,000$ sequences. In addition, we introduce a small amount of white noise, approximately 0.005 dBm, in sequences where the transmit power remains completely flat.

We split the data into three subsets: training ($\sim 9\,000$ sequences), validation ($\sim 4\,000$ sequences), and test ($\sim 4\,000$ sequences). The training subset is used to fit the feature extraction parameters, including PCA, DCT, and the AE model, and to train the validation layers, such as KDE and IFOR. The validation subset is also used for hyperparameter estimation, and the final assessment is performed on the test set. Since the dataset lacks faults, we synthetically introduce them at test time by adding step functions, negative exponentials, or white noise. We set the change-point at $\tau = 750$. The magnitude of each change is defined as $M_c = c \cdot \sigma_s$, where σ_s represents the standard deviation of the sequence. In the case of the negative exponential, the change is described by a time constant T_c.

Cisco Performance Measures

The *Cisco* setup includes twelve transmitted channels, seven of which span 1200 km, while the remaining five covers only 600 km. During this setup, we gathered stable transmission data over several days, taking samples every 30 seconds. Specifically, we collected the Pre-FEC Bit Error Rate (BER) [13], transmit power, and chromatic dispersion [27].

To generate faulty conditions, we disturbed the system both channel-wise and globally. Channel-wise disruptions were introduced using an aggressor channel (AGG) shifting in frequency, while the amplifier gain was adjusted to produce global changes. This approach enabled us to generate instances of both abrupt and gradual changes. Additionally, we received and analyzed optical data from *Cisco* customers, which were collected directly from real-world deployments in the field. Notice that we used this dataset solely for a qualitative assessment due to the limited data available.

Synthetic Datasets

We generate multivariate data streams of dimension d and compute post-change distributions using the *Controlling Change Magnitude* framework [14], which enables us to control the magnitude of the distribution changes. We begin with Gaussian distributions ϕ_0 characterized by a random covariance matrix. Post-change distributions are then generated by applying random rotations and translations to ϕ_0. The post-change distribution ϕ_1 is expressed mathematically as

$\phi_1 = \phi_0(Q + v)$, where Q is the rotation matrix and v is the translation vector. Both Q and v are selected to maintain a fixed distance between the two distributions, which we quantify using the symmetric Kullback-Leibler divergence $sKL(\phi_0, \phi_1)$. In particular, we can generate the in-control sequences by taking i.i.d. samples from a set of modalities $\{\phi_{0,m}\}_{m=1}^{M}$. Then, we introduce a randomized drift after the change-point τ. Formally:

$$x_t \sim \begin{cases} \phi_{0,m} & \text{if } t \in T_m,\ t < \tau \\ \phi_1 & \text{if } t \geq \tau \end{cases} \tag{3.4}$$

were T_m represents randomly generated non-overlapping intervals of the time domain before the change-point.

3.5.2 Figures of Merit

To assess the performance of our framework, we measure the ability to control false alarms while achieving a small delay in detecting the change-point. First, we consider entirely generated from the in-control process $\mathcal{P}_0$ and compute the Average Run Length (ARL_0) and the True Negative Rate (TNR). The $\text{ARL}_0 = \mathbb{E}_{\phi_0}[t^*]$ is the average time elapsed before a false alarm is raised. The TNR is defined as the proportion of sequences that did not trigger any alarms. Notice that we consider the TNR because our test sequences are real acquisitions with finite lengths. Therefore, our method can reach the end of the sequence without raising any alarms.

Moreover, we consider sequences with changes to the out-of-control state and compute the detection delay (DD) and the False Positive Rate (FPR). The DD $= \mathbb{E}[t^* - \tau]$ is the average distance between the detected change-point t^* and the true change-point τ. The FPR is defined as the total number of false alarms raised ($t^* < \tau$) divided by the number of sequences.

3.5.3 Discussion

Pre-processing

When working with individual time series, we fit an AutoRegressive Moving Average (ARMA) model [38] of order (p, q) to capture the underlying structure of the data. To determine the parameters p and q, we employ the Autocorrelation Function (ACF) and the Partial Autocorrelation Function (PACF) [38], specifically identifying the points where the PACF and ACF drop to zero.

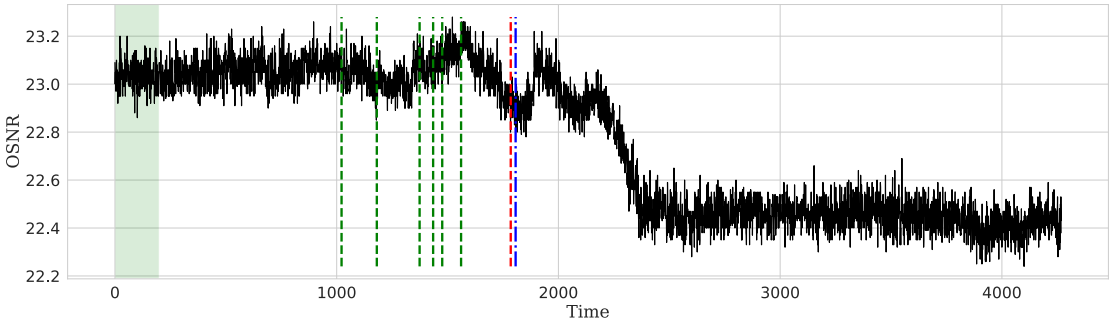


Figure 3.4: OSNR acquired in *Cisco* laboratories. The green dashed lines represent the candidate changes filtered out by our framework. The red dashed lines represent the ground-truth and predicted change-point indices, respectively. The green area represents the training segment for the ARMA model.

Once we have established the order (p, q), we estimate the model's coefficients through Maximum Likelihood Estimation (MLE) [7], given an initial reference segment. An example is depicted in Fig. 3.4, where the green area on the left-hand side highlights the training segment. This process identifies the parameter values that maximize the likelihood of the data. In the multivariate scenario, better results are achieved using a Vector Autoregressive (VAR) model [61]. This model effectively captures the relationships between multiple time series as they change over time. We apply VAR pre-processing when processing the MS-WAN dataset, using the residuals as input for the DL.

Monitoring Real-World PMs

Due to the limited number of faulty PMs acquired in the *Cisco* laboratories, we conducted only a qualitative assessment. An example is illustrated in Fig. 3.5 where HiCAD, with the Normal Discrepancy at the DL and the KDE at the VL, effectively detects deteriorations in fiber links, achieving small detection delays while minimizing false alarms.

We perform the quantitative experiments on the MS-WAN dataset. We compute the ARL_0 and TNR within sequences generated from the in-control process. Conversely, we compute the FPR and the DD in sequences with change-points. In particular, ARL_0 and TNR are averaged over 4400 test sequences with no changes and an average length of 2409.86. Since the MS-WAN dataset does not contain faults, we introduce synthetic changes at $\tau = 750$ using values of $c = 5.0$ and $T_c = 15.0$, as described in Sec. 3.5.1. The experimental results, reported in Table 3.1, evaluate the trade-off between controlling false alarms (in terms of ARL_0

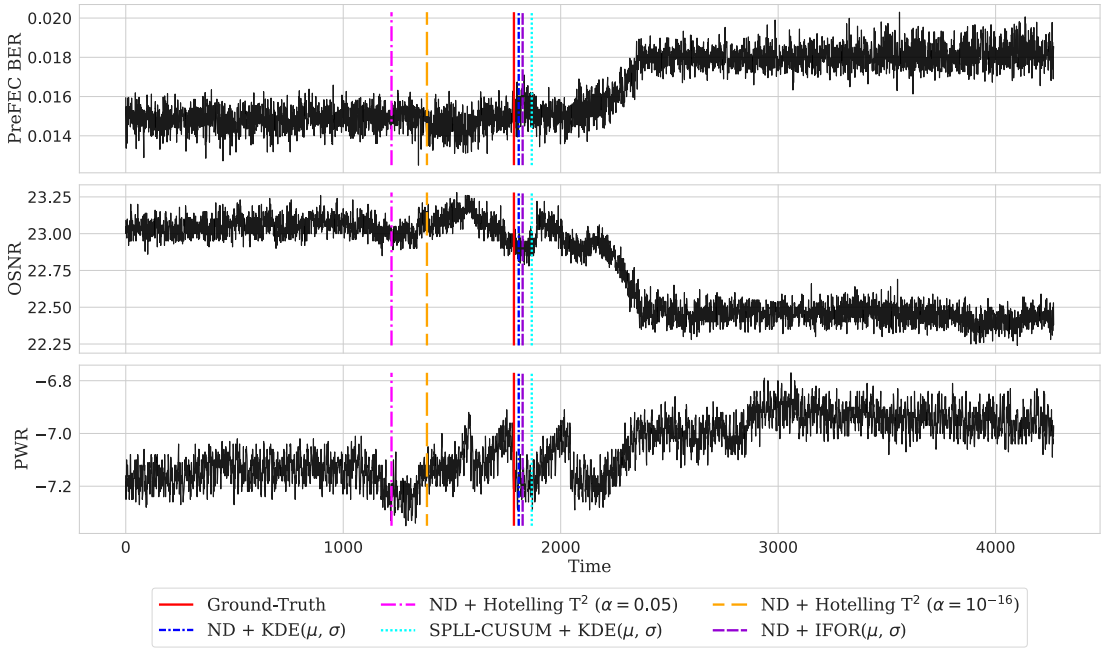


Figure 3.5: Qualitative results of monitoring *Cisco* PMs with our method.

and FPR) and achieving a low DD. Since ND and SPLL-CPM are the two best algorithms in this scenario, we use them within the DL in the following analysis.

First, we show that traditional Change Detection Tests (CDTs), such as SPLL-CPM, SPLL-CUSUM, ScanB, and ND, perform poorly in this context. This is expected because they tend to raise alarms for statistical changes generated in the in-control process, resulting in a high FPR and low ARL_0. Then, we consider HCDT, which validates changes hierarchically using the Hotelling T^2 test at the VL. The ND identifies candidate changes at the DL. HCDT with significance level $\alpha = 10^{-3}$ achieves a FPR of 69.11%. Lowering α to 10^{-10} sets a stricter criterion for detecting changes and reduces the FPR to 29.8%, but causes the FNR to rise significantly from 3.89% to 35.61%. Similarly, when setting $\alpha = 0.05$, making Hotelling extremely sensitive, HCDT achieves a very low FNR of 0.23%, but this comes at the cost of an exceptionally high FPR of 80.39%. These results indicate that, despite the fine-tuning of α, a statistical test at the VL is ineffective in this scenario. While adopting a stricter significance level may reduce the FPR, it simultaneously increases the FNR.

In contrast, HiCAD pairs the DL with a data-driven VL performing significantly better than traditional CDTs. In particular, when using the ND with a trained KDE model based on mean μ and variance σ^2, it yields an FPR of 2.32%

while achieving the lowest detection delay. When KDE is built on PCA and DCT descriptors, it achieves even lower FPR values. However, this improvement comes at the cost of a high FNR, indicating that some actual changes to the out-of-control process may be missed. The ConvAE-based KDE approach achieves an FPR of 1.00%, the lowest in the class of KDE-based VL.

Similarly, with IFOR at the VL, HiCAD achieves satisfactory performance, particularly with PCA and DCT feature extractions. These methods achieve low FNRs (0.25%) and competitive TNR values, although the DD and FPR are slightly higher compared to KDE-based methods. Notice that IFOR, unlike the KDE, does not require a validation dataset to estimate thresholds. When setting SPLL-CUSUM at the DL, we achieve competitive performance, particularly when combined with IFOR. The combination of SPLL-CUSUM with IFOR(μ, σ^2) yields the best TNR (98.10%) and lowest FPR (0.70%) of all methods, making it highly effective in filtering out statistical changes which are part of the in-control state. However, this method also exhibits a high FNR (32.37%), indicating that while it is highly effective in reducing false alarms, it may miss a significant proportion of change-points. KDE-based SPLL-CUSUM methods, particularly with PCA or DCT, demonstrate a more balanced trade-off between DD and FNR, with moderate ARL_0 values and relatively low FPR. Moreover, we present examples of reconstruction-based VL obtained through PCA, DCT, and Autoencoders (AE). PCA and DCT perform better than traditional CDTs. Conversely, the AE provides a good reconstruction of the input window even when it includes changes to the out-of-control state, leading to a large FNR. However, reducing the model complexity hinders the AE's ability to capture even normal fluctuations.

Monitoring Synthetic Data

We generate data streams with dimension $d = 2$ by sampling from three different Gaussian distributions, each having random covariance matrices and mean values. Each sample is drawn independently, but the underlying Gaussian distribution switches randomly, on average once every 300 samples. To train the anomaly detector in the VL, we generate $n_s = 2000$ sequences of length $l = 1000$ that contain no change-points. The DL processes these sequences, and the EL extracts training windows and descriptors for each candidate change. Additionally, we generate $n_s = 1000$ validation sequences of length $l = 1000$ to estimate the thresholds for KDE. To evaluate the control of the false alarms, we assess the empirical ARL_0 and the TNR by generating $n_s = 5000$ test sequences without change-points, each of length $l = 10000$. To evaluate the detection power of the competing methods, we generate $n_s = 5000$ test sequences, each of length

	w/o change		*w/ change*		
Method	**ARL$_0$** ↑	**TNR** ↑	**DD** ↓	**FPR** ↓	**FNR** ↓
SPLL-CPM	188.47	0.20	5.20	95.75	0.0 -
SPLL-CUSUM	327.86	5.11	64.36	79.93	0.23
ScanB	99.03	0.82	0.0	99.18	0.82
ND	243.70	2.55	1.46	87.43	0.0
HCDT [ND + T$^2(\alpha = 0.05)$]	297.86	4.93	32.35	80.39	0.23
HCDT [ND + T$^2(\alpha = 10^{-3})$]	392.87	10.61	147.81	69.11	3.89
HCDT [ND + T$^2(\alpha = 10^{-10})$]	722.54	46.27	347.02	29.80	35.61
HiCAD[ND + KDE(μ, σ^2)]	1136.92	88.82	13.46	2.32	2.34
HiCAD[ND + KDE(PCA)]	1714.30	92.11	75.20	1.52	8.48
HiCAD[ND + KDE(DCT)]	1763.40	92.41	75.35	1.48	8.43
HiCAD[ND + KDE(ConvAE)]	1899.49	93.77	107.49	1.00	13.57
HiCAD[ND + IFOR(μ, σ^2)]	1347.16	83.77	26.21	5.66	3.61
HiCAD[ND + IFOR(PCA)]	1231.28	76.68	24.21	7.39	0.25
HiCAD[ND + IFOR(DCT)]	1251.41	77.50	24.14	7.23	0.25
HiCAD[SC + KDE(μ, σ^2)]	972.79	75.12	44.75	10.49	4.60
HiCAD[SC + KDE(PCA)]	1102.76	78.12	79.05	7.49	3.80
HiCAD[SC + KDE(DCT)]	1132.25	78.32	82.18	7.49	4.20
HiCAD[SC + IFOR(μ, σ^2)]	1183.26	98.10	129.18	0.70	32.37
HiCAD[SC + IFOR(PCA)]	1082.54	80.42	60.48	7.29	6.89
HiCAD[SC + IFOR(DCT)]	1179.22	79.72	55.66	6.89	7.09
HiCAD[ND + Rec(PCA)]	1160.62	87.3	19.52	4.66	4.25
HiCAD[ND + Rec(DCT)]	1160.17	88.59	17.47	4.52	5.30
HiCAD[SC + Rec(ConvAE)]	1384.81	87.57	452.09	4.22	65.11

Table 3.1: Quantitative results on the MS-WAN dataset. Average Run Length (ARL$_0$), True Negative Rate (TNR [%]), False Positive Rate (FPR [%]), False Negative Rate (FNR [%]), and Average Detection Delay (DD) averaged over sequences from the MS-WAN dataset. Normal Discrepancy (ND) or SPLL-CUSUM (SC) are used as Change Detection tests to trigger the different validation layers.

$l = 2000$, with a change-point to the out-of-control state at $\tau = 300$. The post-change distribution ϕ_1 is randomly generated, with the symmetric Kullback-Leibler divergence between the pre- and post-change distributions set at specific

	w/o change		*w/ change* ($sKL = 10$)			*w/ change* ($sKL = 30$)		
HiCAD Method	**ARL$_0$** ↑	**TNR** ↑	**DD** ↓	**FPR** ↓	**FNR** ↓	**DD** ↓	**FPR** ↓	**FNR** ↓
ND + KDE(μ, σ^2)	2059.59	83.10	24.91	2.30	21.70	10.91	2.10	4.20
ND + KDE(PCA)	2375.92	36.00	18.81	7.40	3.20	8.64	5.20	0.00
ND + KDE(DCT)	3080.71	38.50	17.78	3.30	3.60	7.60	2.00	0.00
ND + KDE(ALL)	2936.10	54.40	21.20	3.10	7.10	10.87	1.80	0.30
ND + IFOR(μ, σ^2)	2931.31	16.56	21.30	3.00	1.20	9.01	2.20	0.00
ND + IFOR(PCA)	2189.22	35.88	18.97	7.10	2.20	8.49	5.60	0.20
ND + IFOR(DCT)	2714.93	32.70	18.05	4.30	2.80	7.02	3.80	0.10
ND + IFOR(ALL)	1065.39	0.18	17.27	13.50	0.60	6.95	11.70	0.00

Table 3.2: Quantitative results for the synthetic experiments. ARL$_0$ and TNR are averaged over 5000 sequences of length $l = 10000$ with no changes. DD, FPR, and FNR are averaged over 1000 sequences of length $l = 2000$ with changes of magnitude $sKL = 10$ and $sKL = 30$ at $\tau = 300$.

values, namely $sKL(\phi_{0,i} \to \phi_1) \in \{10, 30\}$. Alongside the DD, we show the FPR and the FNR.

Results are reported in Table 3.2. Methods leveraging PCA or DCT descriptors show lower TNR values but exhibit stronger performance in sequences with changes, suggesting they may be more sensitive to detecting subtle deviations. Methods combining multiple features, i.e., concatenating (μ, σ^2), PCA, and DCT, generally show balanced performance across all the metrics. Indeed, ND + KDE(ALL) achieves a relatively small DD, 21.20 for sKL = 10 and 10.87 for sKL = 30, with low FPR and satisfactory FNR. This suggests that leveraging multiple features can provide a more reliable detection framework across different change magnitudes. Similarly, ND + IFOR(PCA) and ND + IFOR(DCT) exhibit comparable performance, with satisfactory ARL$_0$ and DD for both sKL = 10 and sKL = 30. These results demonstrate that the VL enhances control over false positives without missing true changes, filtering out the fluctuations that are part of the in-control state. Finally, when we focus only on the DL, we obtain an ARL$_0$ of 690.54 with the ND. In contrast, using SPLL-CUSUM yields an ARL$_0$ of 516.93. This demonstrates that combining the DL and VL significantly increases the ARL$_0$ values, enhancing performance by up to 6x.

Computational Time

We conducted our experiments on a device powered by an *Intel Core i5* CPU running at 1.8 GHz. We first measured the execution times of different CDTs:

ND was the fastest at 0.035 seconds, followed by SPLL-CPM at 0.441 seconds, SPLL-CUSUM at 0.310 seconds, and ScanB, which was the slowest, taking 1.797 seconds. Then, we evaluated HCDT by combining Hotelling with ND, resulting in a slightly increased execution time of 0.0375 seconds. Finally, we tested our framework. Specifically, we ran ND + KDE, SPLL-CPM + KDE, and SPLL-CUSUM + KDE, yielding execution times of 0.228 seconds, 1.444 seconds, and 1.511 seconds, respectively. The execution time increases because the method correctly filters out changes that are part of the in-control state, requiring it to analyze a larger portion of the data stream. However, it remains within a range compatible with the system's operation. Notably, the combination of ND + KDE is faster than the SPLL-CPM, SPLL-CUSUM, and ScanB tests, which are commonly used in change detection. This confirms that our framework is suitable for all the tasks where these methods are typically applied.

Limitations

Due to the data-driven nature of HiCAD, estimating a priori thresholds for controlling false alarms is challenging. Its multi-layer architecture complicates the ability to impose direct control over the ARL_0 explicitly. Furthermore, since we do not assume the data to be independent and identically distributed (i.i.d.), standard techniques for controlling false alarms are not applicable. Despite this limitation, the strength of our approach lies in its ability to adapt across various monitoring settings without relying on fixed thresholds or expert-defined rules.

3.6 Conclusions

We introduced a novel framework, *Hierarchical Change and Anomaly Detection* (HiCAD), designed to monitor data streams in non-stationary environments, where distinguishing changes to the out-of-control state from changes within the in-control state is required. HiCAD includes a CDT-based Detection Layer to identify statistical changes, an Embedding Layer to build descriptors that capture their patterns, and a data-driven Validation Layer that successfully detects change-points to the out-of-control state, thereby reducing false alarms. Through rigorous experimentation on real-world datasets, such as optical PMs from *Cisco* and *Microsoft*, we demonstrated that our hierarchical structure provides superior results, enhancing the robustness of the change detection process while maintaining timely responses to relevant system faults. The method that yields better results includes Normal Discrepancy with Kernel Density Estimation and macroscopic statistics.

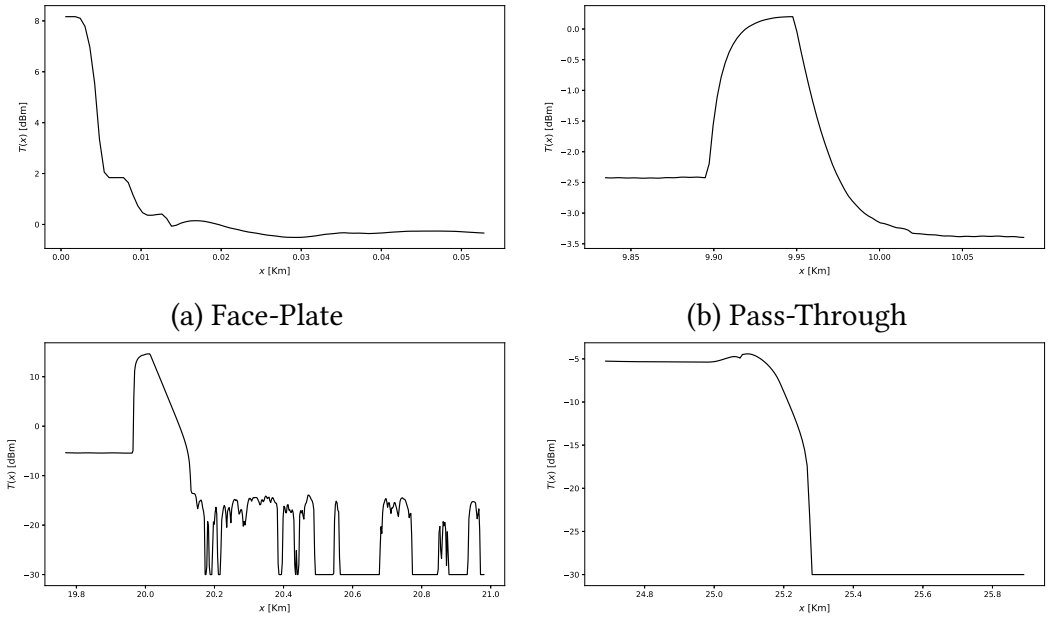
Chapter 4

OTDR Event Detection

Optical Time-Domain Reflectometer (OTDR) traces graphically represent the relationship between optical power and distance along a fiber, revealing characteristic patterns, known as *events* (see Fig. 4.1), that indicate issues such as faulty connectors or fiber breaks requiring corrective action. To enable automatic and accurate diagnostics, we designed the first 1D *Event-Detection Network* learned end-to-end to detect events within an OTDR trace. Furthermore, our solution can identify events that do not belong to any known category or are absent from the training data. Specifically, unknown events are detected through a preprocessing pipeline that identifies local peaks in the OTDR trace. Then, fixed-size windows are cropped around these peaks and classified using an open-set classifier based on OpenMax [6]. This process isolates unknown event regions, which are reported separately. This capability is essential, as these unknown events could indicate more serious issues, yet may be discarded or misclassified under existing categories [28].

Although capable of being trained end-to-end to detect known and unknown events, the proposed *Event-Detection Network* has a significant limitation: it only works on OTDR traces obtained using the same type of device as the one used to collect the training data. When applied to traces from different types of devices, its performance deteriorates due to variations in acquisition technology, which strongly affect the shape of detected events.

An example of this issue is shown in Fig. 4.2, where OTDR traces from two different devices, α and β, are connected to the same fiber link. Both traces display two peaks corresponding to physical connectors along the fiber link (highlighted in yellow). However, despite representing the same fiber span, the traces differ significantly in magnitude and scaling. In particular, the events from device β appear smaller and more compressed, likely due to its lower sampling frequency



(a) Face-Plate

(b) Pass-Through

(c) Fiber-Cut

(d) Fiber-End

Figure 4.1: Examples of OTDR events.
Instances of OTDR events windows in the training set.

in that region. These differences arise from the physical configuration of the OTDR devices used to capture the signal. Consequently, the *Event-Detection Network* trained end-to-end on traces from device α (source domain) may fail to recognize relevant features in traces from device β (target domain).

The lack of feature invariance across diverse types of devices limits the portability of the *Event-Detection Network* from one device to another. As a result, the network needs to be retrained for each specific device, which requires time-consuming data preparation. To overcome this challenge, we present a novel *Event-Detection System* designed to adapt to different devices by leveraging expert-driven algorithms that localize events based on their morphology. Specifically, we introduce the *Interval Proposal Algorithm* (IPA), which uses signal processing techniques and robust fitting methods to identify candidate events along with their *context features*, namely domain-specific information such as power intensity and relative position along the OTDR trace. For classification, we standardize the events and train a feature extractor f_θ to focus on *morphological* features, capturing the shape of the events independently of their context. These domain-invariant morphological features are then combined with the context features produced by the IPA and fed into a classifier $\mathcal{C}$ to predict the event type.

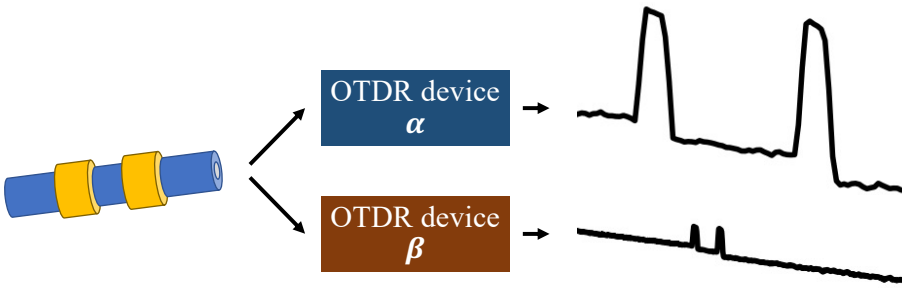


Figure 4.2: Example of domain shift between OTDR traces from different types of devices. On the left-hand side, a fiber link with two connectors. The PASS-THROUGH event signatures on the right-hand side generated by devices of type α and β show substantial differences both in terms of the number of samples and power intensity.

By combining both context and morphological features, our approach ensures accurate event detection across different devices.

Our experimental evaluation, using real-world OTDR traces from multiple fiber links acquired by two Cisco devices – NCS1001 [23] (device α) and NCS1010 [21] (device β) – demonstrates the effectiveness of our methods. Quantitative testing shows that our approach achieves mean average precision mmAP@0.5 of 75.33% for traces from device α, significantly outperforming the 69.27% achieved by the 1D Faster R-CNN. For traces from device β, our method scores 64.93% without fine-tuning, while the 1D Faster R-CNN fails to detect events. These results validate that our method provides a robust OTDR event-detection network adaptable to various devices. Additionally, our leave-one-class-out validation approach confirms that the network's backbone can effectively detect unknown events.

4.1 Problem Formulation

An OTDR trace is represented as a sequence of n samples $T = \{(x_1, p_1), (x_2, p_2), \dots, (x_n, p_n)\}$, where $p_i \in \mathbb{R}$ is the reflection loss (in dBm) measured at distance $x_i \in \mathbb{R}$ (in km) from the fiber's beginning. A trace T may contain multiple events, each one described by a triplet $e = (y, x^s, x^e)$, where x^s and x^e are the start and end points of the portion of the trace containing the event of class y. We assume to have a fully labeled training set $\text{TR}^{\mathcal{S}} = \{(T_j, E_j), j = 1, \dots, N\}$, containing N traces from the source domain

$\mathcal{D}^{\mathcal{S}}$, where $E_j = \{e_i\}_{i=1}^{M_j}$ is the set of events in trace T_j. Due to the unique nature of optical events in OTDR traces, the training set TR might not cover all the possible event types, thus the label set $Y = Y_K \cup Y_U$ is the union of known events Y_K, represented in TR, and unknown events Y_U, which might appear only during testing. Notice that $Y_K \cap Y_U = \emptyset$. Furthermore, we consider two OTDR devices, α and β, each implementing different sampling and post-processing techniques, resulting in traces belonging to source domain $\mathcal{D}^{\mathcal{S}}$ and target domain $\mathcal{D}^{\mathcal{T}}$, respectively. In particular, events collected from domain $\mathcal{D}^{\mathcal{T}}$ might have different power intensities and widths, and an event-detection network trained on traces from $\mathcal{D}^{\mathcal{S}}$ might be unable to generalize to events from $\mathcal{D}^{\mathcal{T}}$. Due to limited data availability, fine-tuning a model specifically for the target domain $\mathcal{D}^{\mathcal{T}}$ is not feasible.

Our goal is to design a method, trained exclusively on the known annotations in Y_K from domain $\mathcal{D}^{\mathcal{S}}$, that can automatically detect all events in a trace T, regardless of whether they originate from $\mathcal{D}^{\mathcal{S}}$ or $\mathcal{D}^{\mathcal{T}}$. We formulate this as a *Domain Adaptation* (DA) problem, meaning we aim to compensate for the shift between training and test data distribution. Whenever the trace T contains an unknown event – whose label belongs to Y_U – the method localizes it and labels it as an *unknown event*.

In our study, we consider four types of known events, as reported in Fig. 4.1:

- Face-Plate denotes the fiber start and determines the location at which the OTDR tool injects its pulses.
- Pass-Through denotes the mechanical conjunction of two fiber links.
- Fiber-End denotes the endpoint of the fiber link.
- Fiber-Cut denotes the abrupt termination of the fiber induced by a cut.

As customary in detection networks, we include also the No-Event label to discard regions that do not contain an event.

4.2 Related Literature

This section reviews the machine learning literature on event detection in OTDR traces. It covers object detection networks (Sec. 4.2.1), deep neural networks for time series analysis (Sec. 4.2.2), open-set recognition methods (Section 4.2.3), deep learning for OTDR traces (Sec. 4.2.4), and domain adaptation (Sec. 4.2.5).

4.2.1 Object Detection Networks

Object-detection networks are designed to both localize and classify objects from a set of predefined classes. These systems are primarily used in Computer Vision to autonomously detect objects within images. Our focus is on networks built on the R-CNN architecture [40, 39, 79], which have inspired the design of the *Event-Detection Network*. R-CNN-based detection networks consist of two main components: the first generates region proposals, and the second performs classification on these proposals.

The original R-CNN [40] introduced an external region proposal mechanism that heuristically identifies potential object areas in an image. These regions are then cropped, resized, and processed by a CNN for classification and localization refinement. Subsequent advancements include Fast R-CNN [39], which improves processing speed compared to R-CNN by directly extracting regions from the network's intermediate representations. Faster R-CNN [79] further enhances this approach by incorporating a Region Proposal Network (RPN) that generates region proposals more efficiently. Faster R-CNN is highly effective, as the RPN can be trained to optimize detection performance. The high accuracy of Faster R-CNN demonstrates that CNN features generate precise region proposals, which the RPN refines by computing the *objectness scores* across various grid locations.

In contrast, single-shot detection methods apply the neural network to the entire image to directly identify objects. YOLO [77] is a notable example of this approach, known for its rapid inference capabilities — it can process video frames in real-time. However, this speed comes at the cost of slightly reduced accuracy compared to Faster R-CNN. YOLO's subsequent versions, such as Yolo9000 [76], offer improvements over the original architecture. Later versions of YOLO have become standard for object detection tasks in computer vision. Other significant single-shot detection networks include SSD [59] and RetinaNet [56].

For the design of our event-detection networks, we draw inspiration from the Fast R-CNN and Faster R-CNN since these architectures are simpler than YOLO's, and we do not have strict timing constraints for OTDR monitoring. OTDR traces are analyzed less frequently than video frames and, being one-dimensional signals, are computationally less demanding than images.

4.2.2 Deep Learning for Time Series

The effectiveness of Convolutional Neural Networks (CNNs) in visual recognition tasks extends to 1D time-series classification [31], including applications to audio recordings, electroencephalograms (EEG), and electrocardiograms

(ECG). ConvTimeNet [47] highlights that fully convolutional neural networks can achieve high classification performance without requiring dimensionality reduction through pooling layers. More recent work, such as InceptionTimeNet [32], shows that deeper CNN models with residual connections can set new benchmarks in time-series classification.

Time-series object detection, which involves localizing and classifying objects within 1D signals, has not been as extensively studied as image-based object detection. However, there are some notable exceptions, such as anomaly detection in ECGs [100], earthquake detection in seismic waves [97], and keyword spotting in audio signals [72]. A recent example is SpeechYOLO [86], which is designed to detect and localize specific keywords or speech elements within fixed-length audio signals. Our event-detection networks employ a similar approach, adapting an image-based object-detection network for time series data. Nonetheless, our models are tailored for the OTDR domain and can detect unknown events.

4.2.3 Open-Set Recognition

Most deep neural networks used for classification are designed for the closed-set scenario, where each input is assigned to one of the predefined labels seen during training. This approach can be too restrictive as test inputs might belong to categories not encountered before. To tackle this issue, Open-Set Recognition (OSR) techniques [83] allow networks to classify the inputs that do not fit any known categories as unknown. To achieve this, methods such as thresholding the posterior probabilities from Softmax or using OpenMax [6] have been developed. More recently, several alternatives to OpenMax have been proposed, incorporating novel learning methods [71, 17].

The application of Open-Set Recognition in object detection has only recently been explored [28, 45, 82]. This lag may be due to the complexity of integrating OSR into detection networks, which typically include a *background* class intended to capture objects that do not belong to known categories. These networks are trained on large datasets of natural images, which usually contain numerous unknown objects in addition to the annotated ones. As a result, unknown objects are often included during training and implicitly associated with the background class [45]. Consequently, there has been limited research on object detection in more realistic scenarios where models must identify unknown objects. Most studies have focused on estimating model uncertainty for each prediction and filtering out low-confidence detections [66, 53].

4.2.4 Deep Learning for OTDR Trace Analysis

Deep learning models are used to analyze OTDR traces and nearby events such as human movement or seismic activity. Recent studies [3, 88, 54] show that these models can classify such events, though initial detection typically relies on an external algorithm or pipeline. For example, [96] introduces a 1D neural network based on sequence learning that processes a de-noised 1D signal to identify intrusion events. Similarly, [87] proposes a 2D CNN classifier for OTDR traces that accurately categorizes events into five types, including walking and digging.

In contrast, the *Event-Detection Network* presented here differs significantly in both application and methodology. Unlike previous networks focusing on seismic data or general events unrelated to optical sensing, our network is specifically designed to detect OTDR events. Furthermore, while previous methods approach event recognition as a purely classification problem, our network classifies and localizes events, estimating their position and extent along the OTDR trace. Importantly, our solution addresses the challenge of detecting unknown events, a problem not covered by existing methods.

4.2.5 Domain Adaptation

Domain Adaptation (DA) is a well-studied problem, and several methods have been proposed. A stream of research learns domain-invariant feature representations. For example, [37] introduces a domain classifier alongside the feature extractor, using a gradient reversal layer to align the feature distributions across different domains.

Another line of research employs Generative Adversarial Networks (GANs) for domain mapping. For instance, [10] transforms source-domain data to mimic the target domain. A similar approach is discussed in [15], where DA is applied to ECG heartbeats. These techniques generally require target-domain data during training, even if it's unlabelled. In contrast, our approach assumes that no target-domain data is available during training.

Additionally, [52] proposes a strategy that adjusts the mean and standard deviation of Batch Normalization layers during inference. However, this method requires modifying the network during inference, which may not be practical in industrial settings or for standalone devices. Instead, our approach involves preprocessing the input trace with expert knowledge and standardizing candidate events before classification. Inspired by [9], which separates image content from style for DA, we distinguish the morphology of OTDR events from their contextual information to improve recognition and detection performance.

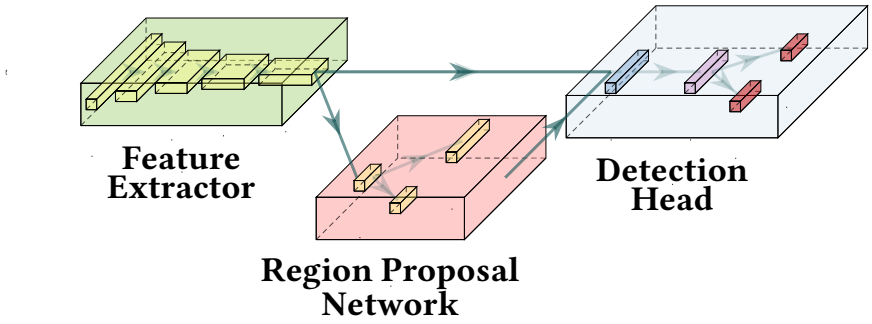


Figure 4.3: Architecture of the proposed Event-Detection Network.

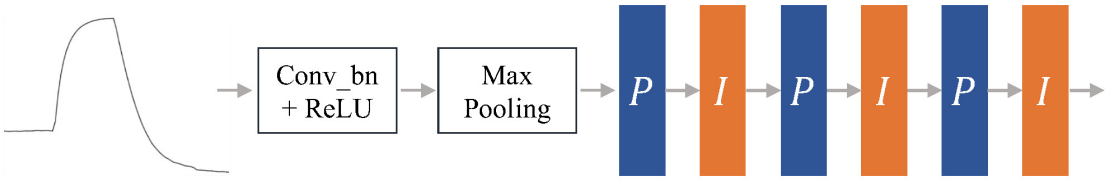


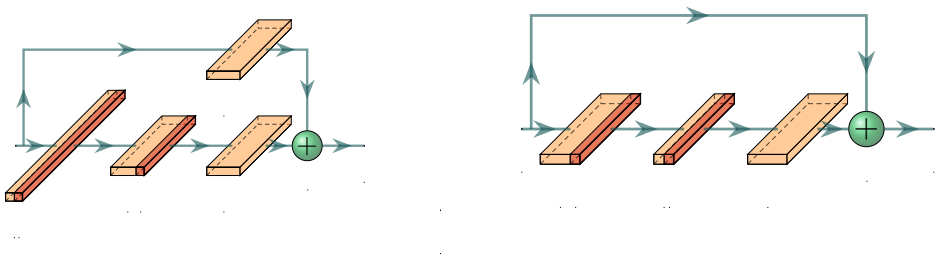
Figure 4.4: Architecture of the Feature Extraction Network. The P blocks colored in blue are the 1D-Conv-Projection block, and the I blocks colored orange are the 1D-Conv-Identity block.

4.3 Event-Detection Network

Our *Event-Detection Network* is based on the Faster R-CNN architecture but has been adapted to process OTDR traces instead of images. The network identifies events and returns them as a set $\{(y, x_s, x_e)\}$, where y represents the label of the event occurring between locations x_s and x_e. Notice that the network autonomously determines the optimal window for each event rather than relying on fixed windows.

4.3.1 Network Architecture

The proposed *Event-Detection Network* is illustrated in Fig. 4.3. It is inspired by the Faster R-CNN architecture [79] and includes three main components: the *Feature Extraction Network* (FEN), the *Region Proposal Network* (RPN), and the *Detection Head*. The FEN consists of 22 convolutional layers with a receptive field of 278 and an effective stride of 8. As shown in Fig. 4.4, its architecture includes *i)* a convolutional layer with 13 filters of size 1×3, followed by *Batch Normalization*, *ii)* a *Max Pooling* layer of size 1×2, *iii)* three pairs of *1D-Conv-Projection* (Fig. 4.5a) and *1D-Conv-Identity* blocks (Fig. 4.5b), with each block consisting of 3 and 4



(a) 1D-Conv-Projection block (b) 1D-Conv-Identity block

Figure 4.5: Building blocks of our network architecture.

convolutional layers, respectively. Despite having 22 convolutional layers, the 1D nature of the FEN keeps it lightweight, with a total of only 83 888 parameters.

The RPN is responsible for localizing candidate events within the trace. It operates on the feature maps extracted by the FEN and generates region proposals based on a set A of anchors, namely predefined segments. The RPN has two output layers and employs a sliding window approach over the FEN's feature maps. For each spatial location and each anchor in A, these layers provide estimates for region proposals. Each proposal includes an eventness score p and an offset t. The eventness score p reflects the likelihood of an event occurring at a specific location within the proposed region, while the offset t adjusts the proposal to better match nearby annotated events. Consistent with [79], the offset for an anchor $a \in A$ is parameterized as $t = (t_{a,x},\ t_{a,w})$, where $t_{a,x}$ is a scale-invariant translation term and $t_{a,w}$ is a log-space scaling term for width:

$$t_{a,x} = \frac{x - x_a}{w_a}; \quad t_{a,w} = \log \frac{w}{w_a} \tag{4.1}$$

where x_a and x represent the locations of the anchor and the estimated region, respectively, while w_a and w denote their widths. In our OTDR event detection system, we use three distinct anchors, resulting in $|A| = 3$ estimates for each location on the feature maps provided by the FEN.

Finally, the Detection Head processes inputs from both the FEN-generated feature maps and the RPN-generated region proposals. These inputs are combined and processed by a Region of Interest (RoI) pooling layer, which produces a fixed-length feature vector for each region proposal. Since region proposals can vary in spatial extent depending on the anchor, the RoI pooling layer standardizes the feature vector length. The resulting output is then fed into two parallel layers: a classification and regression layers. The classification layer consists of a fully connected layer followed by a *Softmax* activation function, generating a

vector of $|Y| + 1$ posterior probabilities for each RoI, including one probability for each class in Y_K and one for the NO-EVENT class. Meanwhile, the regression layer provides $|Y|$ outputs, each representing the adjustment needed to refine the RPN predicted event locations for specific classes. Different adjustments are applied depending on the event class.

Overall, the network has 103 108 parameters, which is significantly fewer than 2D object detection networks.

4.3.2 Training Procedure

The training process for the network follows that of the Faster R-CNN [79]. In particular, we first train the Feature Extraction Network (FEN) on a separate event-classification task using a fixed input size. This initial training helps the FEN learn meaningful features, which then facilitates the training of the Region Proposal Network (RPN) and the Detection Head (DH). Specifically, we apply an *alternating training* strategy [79] following these steps:

1. **Freeze FEN.** Initially, we freeze the weights of the pre-trained FEN and fine-tune the RPN layers for the region proposal task. In particular, we minimize the loss in Eq. (4.3), which ignores event types.

2. **Freeze RPN.** Next, we freeze the RPN and fine-tune both the FEN and the DH on the proposals provided by the RPN. Here we minimize the event detection loss in Eq. (4.4), which considers both event locations and labels.

3. **Freeze FEN and DH.** We then freeze the FEN layers and the DH, focusing on fine-tuning the RPN layers by minimizing the loss function in Eq. (4.3).

4. **Freeze FEN and RPN** Finally, we freeze the FEN backbone and the RPN layers, and then fine-tune only the layers of the DH to minimize the loss in Eq. (4.4).

For training the FEN, we incorporate a *Global Average Pooling* (GAP) layer to enhance translation invariance, followed by a linear layer and a Softmax activation function. We prepare the training set for event classification by cropping fixed-size windows of $2w = 300$ samples around events and no-event examples, which contribute to the background class. The event classification loss in a window W is given by the categorical cross-entropy over the known event types Y_K:

$$\mathcal{L}\left(y, \hat{y}\right) = -\sum_{k=1}^{|Y_K|} y_k \log(\hat{y}_k), \tag{4.2}$$

where y_k and $\hat{y}_k$ represent the ground-truth and predicted one-hot encoded labels, respectively. After training, we remove the GAP and Softmax layers, leaving the trained layers as the FEN backbone for the 1D-Faster R-CNN (see Fig. 4.3).

To address the class imbalance and the limited amount of annotated events, we use several data augmentation techniques during training. In each batch, we randomly select 10% of events and apply a right/left translation of the OTDR trace by a random amount between 5% and 25% of its original size. Additionally, we introduce a random offset within $[-8, +8]$ to 5% of the events in the batch to shift the time series values. Besides these standard augmentations, we use mix-up data augmentation [103], which has proven effective for deep CNNs in various tasks. Mix-up generates new training samples by creating convex combinations of two randomly selected windows and the corresponding linear combination of one-hot encoded labels as targets. Although the mixed events lack physical realism, we find this technique significantly improves the generalization of the *Event-Detection Network*.

For training the RPN, the loss function evaluates how closely the estimated anchors match the annotated events, regardless of their class. Each anchor is assigned an *eventness score* that measures its likelihood of belonging to one of the known categories rather than being a no-event. The loss function for the RPN is given by:

$$\mathcal{L}\left(y^e, \hat{y}^e, t, \hat{t}\right) = \sum_i \mathcal{L}_{cls}\left(y_i^e, \hat{y}_i^e\right) + \lambda \cdot \sum_i y_i^e \cdot \mathcal{L}_{reg}\left(t_i, \hat{t}_i\right), \tag{4.3}$$

where y^e and $\hat{y}^e$ are the ground-truth and predicted eventness scores, while t and $\hat{t}$ are the ground-truth and predicted offsets. The classification loss $\mathcal{L}_{cls}$ is the binary cross-entropy over the event/no-event classes. The terms y_i^e and $\hat{y}_i^e$ represent the ground-truth and predicted eventness scores for the i-th anchor. We set $y_i^e = 1$ if the Intersection-over-Union (IoU) between the i-th anchor and any annotated event exceeds 0.5; otherwise, $y_i^e = 0$.

Following [79], we use the $\mathcal{L}_{1,smooth}$ as the regression loss:

$$\mathcal{L}_{1,smooth}\left(t_i, \hat{t}_i\right) = \begin{cases} \frac{1}{2}\left(t_i - \hat{t}_i\right)^2, & \text{if } |t_i - \hat{t}_i| < 1 \\ |t_i - \hat{t}_i| - \frac{1}{2}, & \text{otherwise.} \end{cases} \tag{4.4}$$

This regression loss is applied to each component of the predicted and ground-truth offsets. It is weighted by y_i to focus on localization errors only for positive anchors. The parameter λ balances the classification and regression components of the loss function. Finally, for each Region of Interest (RoI), the multi-task loss [79] is computed as follows:

$$\mathcal{L}\left(y_i, \hat{y}_i, t_i, \hat{t}_i\right) = \mathcal{L}_{cls}\left(y_i, \hat{y}_i\right) + \lambda \cdot \left[y_i \geq 1\right] \cdot \mathcal{L}_{reg}\left(t_i, \hat{t}_i\right) \tag{4.5}$$

Algorithm 2 Known and Unknown Event Detection

Input: Trace T, Event-Detection Net $\mathcal{E}$, open-set classifier $\mathcal{C}$, window-size w
Output: E_K set of known events detected, E_U set of unknown events detected

1: $E_K = [],\ E_U = []$; ▷ Initialize the output sets
2: $T' = \text{Detrend}(T)$
3: Detect the set of local peaks P in the trace T'
4: **for each** $x_0 \in P$ **do** ▷ Test each candidate event using OSR Classifier $\mathcal{C}$
5: Define the window $W = \{T'(x_0 - w + 1), \dots, T'(x_0 + w)\}$
6: Classify W: $\hat{y} = \mathcal{C}(W)$
7: **if** $\hat{y} ==$ UNKNOWN **then**
8: $E_U = [E_U, (\text{UNKNOWN}, x_0 - w + 1, x_0 + w)]$
9: **end if**
10: **end for**
11: Detect events $E_K = \mathcal{E}(T)$. ▷ Event-Detection Network
12: Remove from E_K any unknown event detected inside E_U.

where y_i and $\hat{y}_i$ denote the ground-truth and predicted classes for each RoI, while t_i and $\hat{t}_i$ are the offsets of the ground-truth and predicted events relative to the anchor. The classification loss $\mathcal{L}_{cls}$ is defined by the categorical cross-entropy in Eq. (4.2). As in Eq. (4.3), λ balances the classification and regression terms of the multi-task loss. The term $[y_i \geq 1]$ equals 1 if $y_i \geq 1$ and 0 otherwise, ensuring that the regression loss is calculated only for event anchors. The regression loss $\mathcal{L}_{reg}$ is defined as in Eq. (4.3).

4.3.3 Detection of Known and Unknown Events

Our proposed solution, detailed in Algorithm 2, detects unknown events by preliminary trace analysis to identify candidate regions. The detection of the known events corresponds to a forward pass through the network $\mathcal{C}$.

First, we perform detrending on the input trace T (line 2) by robustly fitting a line, exploiting the RanSaC algorithm [33]. This step eases the detection of local maxima (line 3). Around each detected peak, we select a fixed-size window W containing $2w = 300$ samples (line 5). This window is then passed to an open-set classifier $\mathcal{C}$ (line 6), described in Sec. 4.3.3.

If the window W is classified as UNKNOWN (line 8), it is added to the set of unknown events E_U. Any other label returned by $\mathcal{C}$ is ignored, as known events are more accurately classified when processed by the entire network. Thus, the original trace T is processed (line 11) to retrieve the known events $\mathcal{E}$. Finally, to

avoid wrong classifications, we remove any event from E_K that overlaps with the support of an unknown event in E_U (line 12).

Open-Set Classifier

Initially, we train a *closed-set* classifier $\mathcal{C}$ on the training set TR with known events Y_K. The classifier $\mathcal{C}$ adopts the architecture of the FEN, which has been demonstrated to be effective. It processes input through a fixed window W containing $2w = 300$ samples. To obtain class posterior probabilities, we incorporate a Global Average Pooling (GAP) layer [55] and a Softmax activation layer at the top of the network.

The classifier $\mathcal{C}$ is then trained to classify windows W that are either centered around annotated events in the training set TR or randomly cropped from portions of the trace that contain no events. The same data augmentations used for training the FEN are applied here, but this time, training is done on detrended traces T' since $\mathcal{C}$ operates after the detrending process. Once $\mathcal{C}$ has been trained, the final layer is transformed into an open-set classifier. Specifically, we implement two open-set classifiers: Softmax thresholding to assess the network's prediction confidence and the *OpenMax* algorithm [6].

Since the Softmax output can be interpreted as a probability distribution over the known classes, a low probability indicates the model's uncertainty in its prediction. As a result, we can set a threshold on the Softmax output, considering any input with uncertain predictions as belonging to the unknown class. In particular, an input is labeled as UNKNOWN when the following condition holds:

$$\max_i(\pi)_i < \tau \tag{4.6}$$

where $\pi \in \mathbb{R}^{|Y_K|}$ represents the class probabilities predicted by the classifier, and τ is a threshold set to reduce false alarms (NFA) in the training set [91].

Conversely, the OpenMax method [6] estimates the probability of being unknown by analyzing the distribution of activation vectors from the training data. For each correctly classified training window $W \in \mathbb{R}^{2w}$, it computes an activation vector $V(W) \in \mathbb{R}^{|Y_K|}$. OpenMax then performs the following steps for each class y: *i*) It groups all activation vectors based on their true class y, creating the set $A_y = \{V(W) \mid \mathcal{C}(W) = y\}$; *ii*) It calculates the average of A_y, producing a mean activation vector $\mu_y \in \mathbb{R}^{|Y_K|}$; *iii*) It computes the distance between μ_y and all other activation vectors in A_y; *iv*) It applies Extreme Value Theory (EVT) to fit a Weibull distribution to μ_y and the δ most distant activation vectors in A_y.

During testing, for a given input W, OpenMax uses the previously fitted

Weibull distribution to adjust the input's activation vector, producing a recalibrated vector $\hat{V}(W)$. The difference between the original activation vector $V(W)$ and the recalibrated vector $\hat{V}(W)$ reflects the network's uncertainty in its prediction. The unknown score U^{W} is then calculated as:

$$U^{\mathrm{W}} = \sum_{i=1}^{|Y_K|} V_i(W) - \hat{V}_i(W) \tag{4.7}$$

Finally, OpenMax appends U^{W} to the recalibrated activation vector $\hat{V}(W)$, effectively introducing an additional class, labeled UNKNOWN. The vector $\left[\hat{V}_i(W) \mid U^{\mathrm{W}}\right]$ is then passed through a Softmax layer, yielding recalibrated probabilities for the known classes, as well as for the unknown class. If the highest probability corresponds to the UNKNOWN class, or if the maximum probability is below a set threshold τ, the input W is classified as UNKNOWN.

4.4 Event-Detection System

Domain shift hinders the portability of the *Event-Detection Network* due to variations in event width and power magnitude across devices. Therefore, the model must be trained on data from the same domain as the testing data, requiring the acquisition of a dataset for each device type. To address this limitation, we developed a novel *Event-Detection System* (see Fig. 4.6) that decouples event localization from classification, enabling the standardization of candidate events to learn domain-invariant features. Furthermore, we combine context information about the event's location within the trace to improve classification accuracy.

The input OTDR trace is first processed by the Interval Proposal Algorithm (IPA) to identify a set of candidate events along with their context features (Sec. 4.4.1). Then, we standardize events (Sec. 4.4.2) and pass them through the Feature Extractor f_θ to extract morphological features, which are then concatenated with the context features. Finally, we feed the concatenated feature vector into the classifier $\mathcal{C}$ to predict the event labels. Details about the network architecture and training phases can be found in Sec. 4.4.3, while Sec. 4.4.4 provides implementation details.

4.4.1 Interval Proposal Algorithm

The Interval Proposal Algorithm (IPA) analyzes the trace T to identify potential events e_i by extracting variable-size intervals (x_i^s, x_i^e). The IPA consists of two main stages: peak detection and interval estimation.

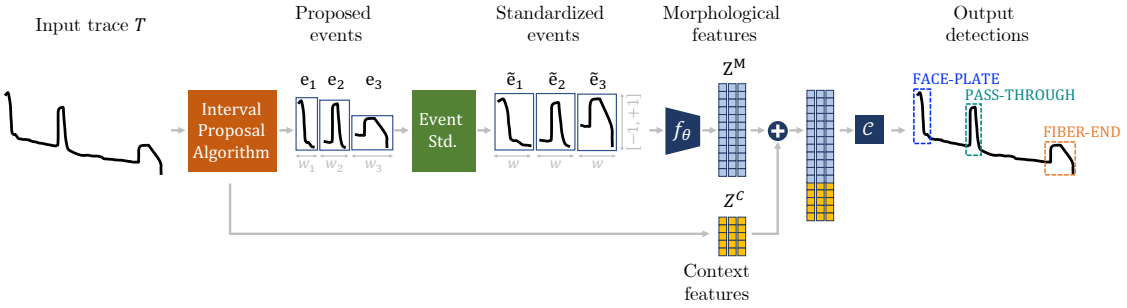


Figure 4.6: The proposed *OTDR Event-Detection System* consists of an Interval Proposal Algorithm that localizes proposed events and extracts context features, such as event power and position, and Event-Standardization module that normalizes events in size and power, and a ResNet-inspired Feature Extractor learned to capture morphological features. The classifier is then trained by combining both morphological and context features.

Peak detection. In the first stage, the IPA identifies candidate event locations $x_i \in \mathcal{P}$ by localizing local maxima $\mathcal{M}$ in the detrended trace, achieved through linear trend fitting with the MSAC method [19]. This stage is divided into two parts to enhance robustness against noise and avoid detecting minor peaks. First, high peaks $\mathcal{P}^h = \{x_i \in \mathcal{M} : \rho(x_i) \geq H\}$ are identified as local maxima that meet or exceed a specified prominence threshold H, indicating their significance relative to neighboring peaks. Then, low peaks are identified as $\mathcal{P}^l = \{x_i \in \mathcal{M} : L \leq \rho(x_i) < H \text{ and } \forall x_k \neq x_i, |x_i - x_k| > d\}$. These peaks have prominence values that fall between L and H and are separated by at least d samples. The overall set of candidate peaks is subsequently denoted as $\mathcal{P} = \mathcal{P}^h \cup \mathcal{P}^l$.

Interval estimation. In the second stage, depicted in Fig. 4.7, the IPA generates two windows of size w on either side of each detected peak $x_i \in \mathcal{P}$, designated as $W_i^l = \{x_{i-1} - w, \ldots, x_{i-1}\}$ and $W_i^r = \{x_{i+1}, \ldots, x_{i+1} + w\}$. If another peak falls within these windows, the cropping is restricted to the position of that peak. Each window is used to fit a trend line, yielding two polynomials, ν^l and ν^r, that describe the surrounding regions. The start of the interval, x_i^s, is defined as the right-most inlier[1] of ν^l, while the end, x_i^e, is identified as the left-most inlier of ν^r. A margin may be added to capture the event's tails more effectively. Additionally, for each candidate event window (x_i^s, x_i^e), the IPA extracts context features z^C, which include the minimum, average, and maximum power

[1] An inlier is a sample that lies closer to the model than a threshold ε.

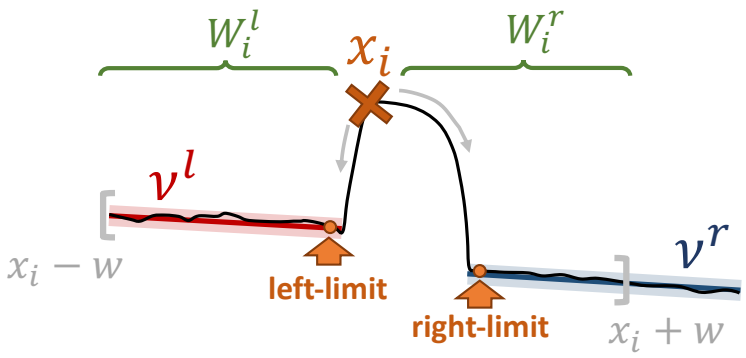


Figure 4.7: Estimation of the window interval for a candidate OTDR Event.

within the localized event window, as well as the peak's location x_i in kilometers. To further improve the classification, since certain events often occur at similar relative positions, we can enhance the context feature z^C by embedding the cosine-based positional encoding [90]. Finally, the IPA successfully returns event-window intervals (x_i^s, x_i^e) and their context features.

4.4.2 Event Standardization

Event standardization is essential for capturing the morphology of events, enabling us to extract features that remain invariant across different devices. To standardize the widths of events originating from different domains, we resize the variable-sized candidate events e_i, identified by the IPA, to a fixed window of w samples. Additionally, to address the differences in power intensity observed among various devices, each event is normalized to fall within the range of $[-1, 1]$. The standardized events, represented as $\tilde{e}_i$, are then fed into the feature extractor to derive the morphological features. This process ensures that the model can generalize effectively across different domains.

4.4.3 Network Architecture and Training

Our feature extractor, denoted as f_θ, is inspired by the ResNet architecture [43]. It consists of 28 convolutional layers, a receptive field of 446, and contains only 372 352 parameters. The model takes normalized event windows of size w as input. To enhance generalization across events with significant width variability, we leverage resampling as a form of data augmentation. Specifically, we interpolate event samples to create windows that can be up to 50% longer or shorter.

When down-sampling an event, we apply padding to maintain the window size, while up-sampling necessitates cropping the event.

The training procedure is exclusively conducted on data from the source domain $\mathcal{D}^{\mathcal{S}}$ and consists of two distinct phases. In the first phase, the feature extractor f_θ is combined with an auxiliary linear classifier $\mathcal{C}_0$ for training. The classifier $\mathcal{C}_0$ facilitates learning useful representations from the input data but is discarded after this phase. The feature extractor specializes in extracting what we term *morphological features* z^M, as it is trained on standardized windows that emphasize the shapes of the events.

During the second phase, the feature extractor is frozen, and the classifier $\mathcal{C}$ is trained on top of it. The classifier $\mathcal{C}$ takes as input both the morphological features z^M produced by f_θ and the context features z^C obtained from the IPA. By integrating morphological and context features, classifier $\mathcal{C}$ effectively combines intrinsic event shape information with context-related data, resulting in a more comprehensive feature representation and improved classifier performance.

4.4.4 Implementation Details

The preprocessing stage involves removing segments of the trace where samples significantly deviate from the primary descending trend, as these low-power areas are mostly composed of noise. After this, padding is applied to both the left and right edges to ensure that events near the boundaries are effectively centered within the trace.

For peak detection, prominence values are tailored according to the trace's domain. A lower prominence value of $L = 0.3$ is used for traces in $\mathcal{D}^{\mathcal{S}}$, while a value of $L = 0.1$ is assigned to traces in $\mathcal{D}^{\mathcal{T}}$, where the events are generally smaller. The higher prominence value, H, and the inlier threshold, ε, are both physically interpretable and can be easily tuned. In our experiments, they are set to $H = 2.0$ dB and $\varepsilon = 0.1$ dB, respectively.

4.5 Experiments

In this section, we evaluate the proposed OTDR *Event-Detection Network* for detecting both known and unknown events and the *Event-Detection System* for event detection across devices. First, we provide details about the dataset used (Sec. 4.5.1) and the performance metrics employed in our analysis (Sec. 4.5.2). Then, we analyze the classification performance to recognize OTDR events (Sec. 4.5.3. Additionally, we present the detection performance of the *Event-Detection Network* for both known and unknown events (Sec. 4.5.4). Finally, we

analyze the *Event-Detection System*, reporting both the localization performance of the IPA and the overall detection pipeline (Sec.4.5.5).

All our networks were trained using the Adam optimizer with a learning rate of 10^{-3}, across 200 epochs for each fold, and with $\lambda = 5$ in Eq. (4.3) and Eq. (4.5). The receptive field for the FEN and the classifier $\mathcal{C}$ is 278. To mitigate the risk of overfitting, we employed early-stopping criteria in all experiments.

Regarding execution time, inference with our *Event-Detection Network* takes 5.347 seconds on the NCS1001, the embedded device where the network is deployed. This device features an *Intel Atom® C2718* CPU running at 1.99 GHz and utilizes TensorFlow 1.15. The inference time for our *Event-Detection System* is 3.637 seconds on the same hardware.

4.5.1 Considered Datasets

The dataset of OTDR traces used to validate our method was collected from Cisco facilities over extensive spans of optical fiber, where various optical events were generated by different devices connected at multiple locations along the fiber.

OTDR recordings are stored in the *SOR* file format [69], which contains not only the raw measurements but also information about the OTDR module and the tested link. However, for event detection, we focus solely on the time series of raw measurements, which include the power readings and the corresponding locations of each measurement. Each trace has been annotated to mark the initial and final points of each event, with labels assigned in Y_K using a specialized annotation tool developed for this purpose.

In total, we collected 635 traces from the NCS1001 device (α), which serves as the source domain $\mathcal{D}^{\mathcal{S}}$, and 31 traces from the NCS1010 device (β), constituting the target domain $\mathcal{D}^{\mathcal{T}}$. It is important to note that only the traces from the NCS1001 are used for training.

This dataset does not include any unknown events. To evaluate detection performance on unknown events in real-world traces, we collected two OTDR traces from customers that feature a new event termed Bulk Attenuator, which are solely for testing purposes. Additionally, to comprehensively assess the open-set classifier, we used 1D time series from a completely different domain: ECG tracings from a public dataset [41]. We chose heartbeats because they also exhibit peaks like OTDR events, but their shape is distinctly different, making them suitable for unknown events. To address the intensity differences between heartbeats and OTDR traces, we scaled each heartbeat so that the average peak of all 633 heartbeats equals K times the average peak of the OTDR events. To investigate the performance of the open-set classifier, we considered various values of

$K \in \{0.125, 0.25, 0.5, 1, 2, 4, 8\}$. Higher values of K lead to noticeable changes, while values of K approaching 0 make changes nearly imperceptible.

4.5.2 Figures of Merit

For classification performance, we resort to standard metrics such as accuracy, precision, recall, F_1 Score, ROC-AUC, False Positive Rate (FPR), and False Negative Rate (FNR). Notice that these metrics are also used to evaluate the open-set classifier. Specifically, we consider a detected event a *True Positive* when its IoU with the ground-truth is greater than 0.5, and the label is correctly estimated. *False Negatives* refer to ground-truth events that were not detected, while *False Positives* represent detected events that do not correspond to any ground-truth event (i.e., NO-EVENTs). In addition to the above metrics, we use confusion matrices to better inspect the sources of errors for the different models.

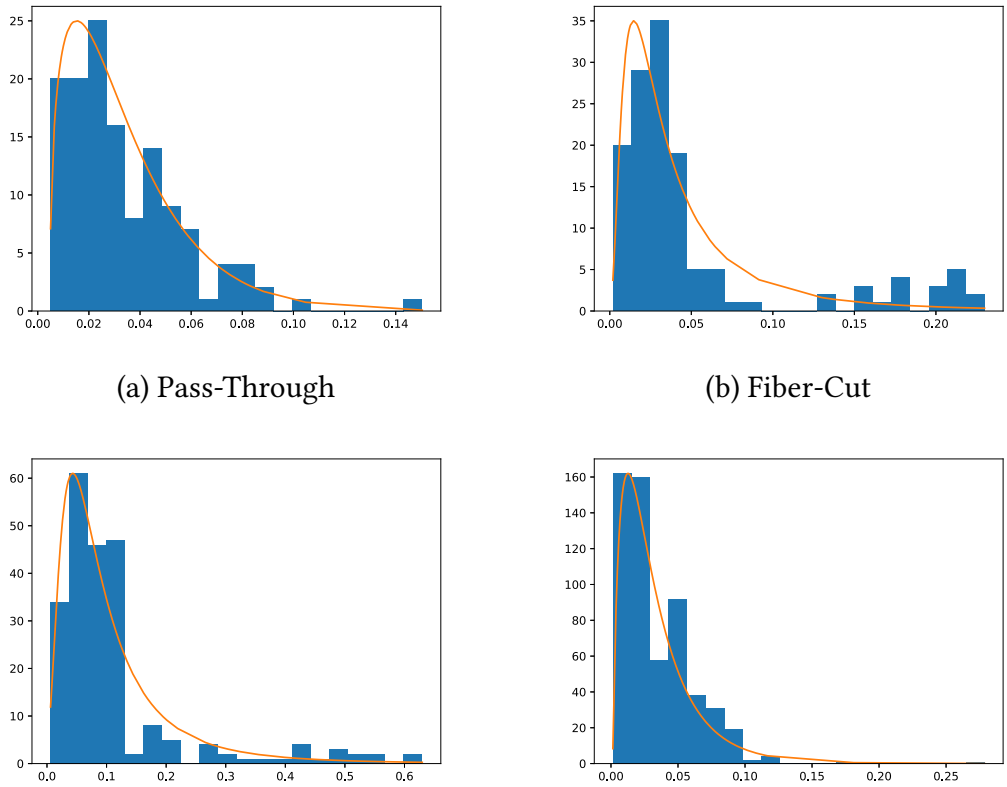
Due to the relatively small size of the dataset, we evaluate the above figures of merit in a 5-fold cross-validation framework. Detection performance is evaluated using object-detection metrics, specifically the mean average precision (mAP) score, a classification and localization accuracy measure introduced in the PASCAL VOC challenge [30]. We also adopt the COCO [57] metric, denoted as mAP@$[.5 : .05 : .95]$: which evaluates mAP at 10 different intersection over union thresholds. Despite both metrics being specifically designed for $2D$ boxes, their adaptation to line segments is straightforward. Finally, the IPA is evaluated in terms of precision and recall for the retrieved peaks.

4.5.3 Event Classification

We assess classification performance for both known and unknown events. To classify unknown events, we conduct two experiments comparing Softmax thresholding with OpenMax. In the first experiment, we systematically exclude specific OTDR event categories from the training set to evaluate the model's ability to recognize these event classes during testing. In the second experiment, we inject ECG signals directly into the traces during the testing phase. Notice that, to exploit OpenMax, we first verified that the distance of the activation vectors from their respective class centroid follows a Weibull distribution, as illustrated in Fig. 4.8.

Known Event Classification

We begin our analysis by examining the classification results of known events. The confusion matrix illustrated in Fig. 4.9a demonstrates that our classifier



(a) Pass-Through

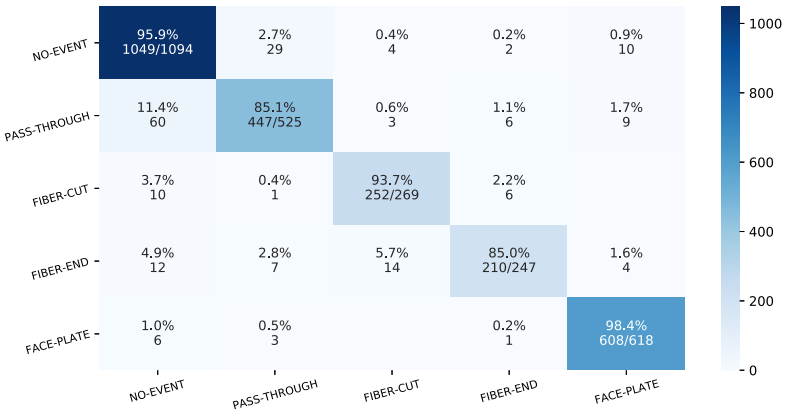
(b) Fiber-Cut

(c) Fiber-End

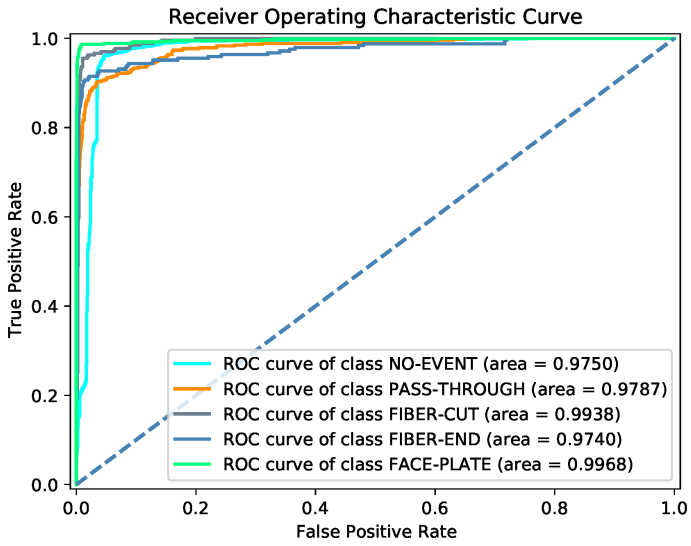
(d) Face-Plate

Figure 4.8: Distribution of the activation vectors' distances from the corresponding class's centroid. As shown in [6], they follow a Weibull distribution.

effectively identifies the OTDR events. We have almost perfect detection for NO-EVENT and FACE-PLATE categories. By visual inspection of the errors, we see that incorrectly classified PASS-THROUGH events often correspond to very low bumps, which tend to be confused for noise and thus labeled as NO-EVENT. A summary of our findings can be found in Table 4.1, where our approach achieves over 90% accuracy on average across various metrics, including precision, recall, F1 Score, and ROC-AUC. The ROC curves presented in Fig. 4.9b further validate the discrimination capability of our model.



(a) Confusion Matrix



(b) ROC-AUC

Figure 4.9: Classification performance of the known events. We report the confusion matrix in Fig. a and the ROC Curve in Fig. b.

Unknown Event Classification with Leave-One-Class-Out

To evaluate the performance of unknown event classification, we systematically exclude one known class from the training set at a time, considering the corresponding OTDR events as unknown. During testing, the classifier is provided with a test-set that includes events from the left-out class. To identify these events as unknown, the classifier is equipped either with Softmax thresholding (with $\tau = 0.75$) or OpenMax (with $\tau = 0.35$ and $\delta = 25$).

	Accuracy ↑	Precision ↑	Recall ↑	F_1 Score ↑	ROC-AUC ↑
Pass-Through	0.851	0.918	0.851	0.883	0.979
Fiber-Cut	0.937	0.923	0.937	0.930	0.994
Fiber-End	0.850	0.933	0.850	0.890	0.974
Face-Plate	0.984	0.964	0.984	0.974	0.997
No-Event	0.959	0.923	0.959	0.940	0.975
Mean	0.916	0.932	0.916	0.923	0.984
Std. Dev.	0.062	0.018	0.062	0.037	0.011

Table 4.1: Classification performance computed by 5-fold cross-validation.

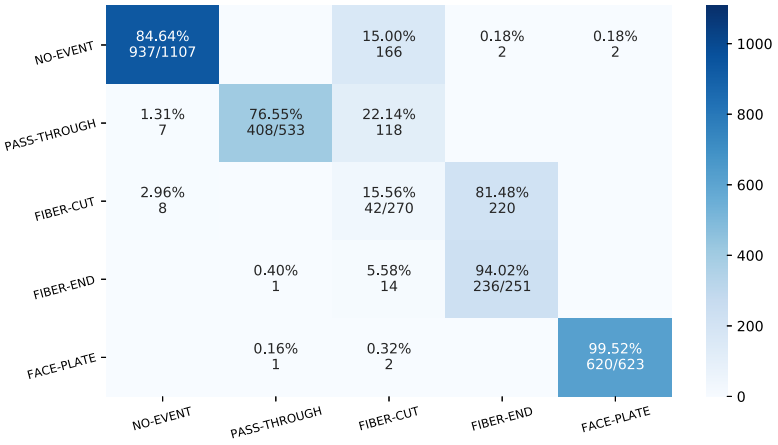
Table 4.2 reports the performance metrics for the unknown class, comparing results from both OpenMax and Softmax thresholding. Each row indicates a left-out class and details the model's performance based on training with the remaining classes.

OpenMax outperforms the Softmax thresholding for all the classes except FIBER-CUT. As illustrated in Fig 4.10, the low performance for the fiber-cut class can be attributed to its confusion with the fiber-end class. These two event categories are similar, and we suspect that without supervision for both, the classifier has not acquired distinctive features to differentiate them, resulting in fiber-cut instances being misclassified as fiber-end. Conversely, when the FIBER-END class is excluded, OpenMax achieves over 74% accuracy. This improvement occurs because the model learns different features when trained on other data, indicating no guarantee of consistent performance when excluding either the FIBER-CUT or FIBER-END classes.

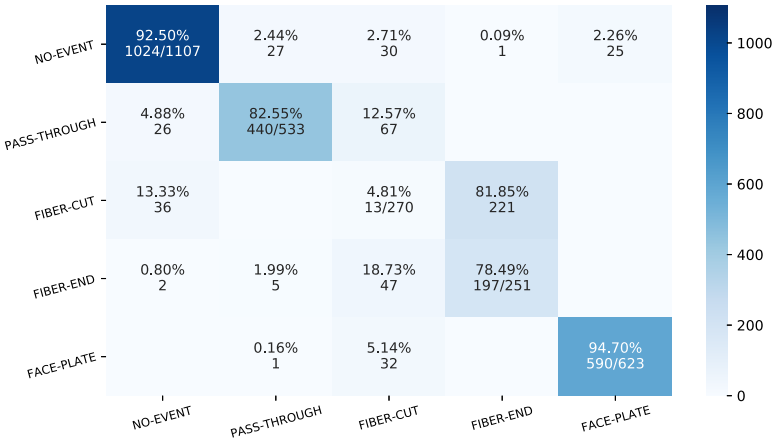
Unknown Event Classification with ECG Tracings

We injected ECG heartbeats into OTDR traces to test whether the classifier could recognize them as unknown. To account for the intensity differences between the heartbeats and the OTDR traces, we scaled each heartbeat so that the average peak of the heartbeats equaled K times the average peak of the OTDR events.

To evaluate the performance of the open-set classifier, we tested various values of K from the set $\{0.125, 0.25, 0.5, 1, 2, 4, 8\}$. Fig. 4.11a illustrates the True Positive Rate (TPR) for both methods as K varies. With $K = 0.125$, the ECG heartbeats are confused with noise and are missed. As K increases, OpenMax effectively identifies a larger number of beats as unknown, ultimately achieving



(a) Softmax thresholding



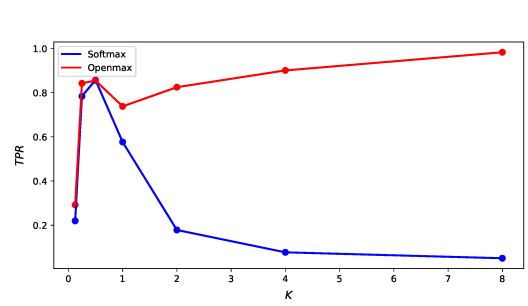
(b) OpenMax

Figure 4.10: Fig. a shows the confusion matrix of the model embedded with the thresholding on the Softmax probabilities. Fig. b shows the confusion matrix of the model embedded with OpenMax.

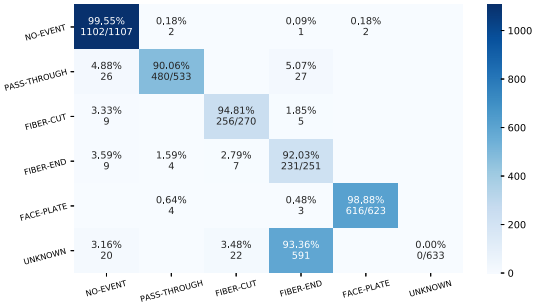
a TPR close to 1.0.

In contrast, Softmax performance decreases as K increases. This is due to its score normalization, which causes ECG beats to become progressively more dissimilar from OTDR windows, hindering the model's ability to extract meaningful features. As a result, the scores provided to Softmax are small and exhibit minimal variation, but appear larger after normalization.

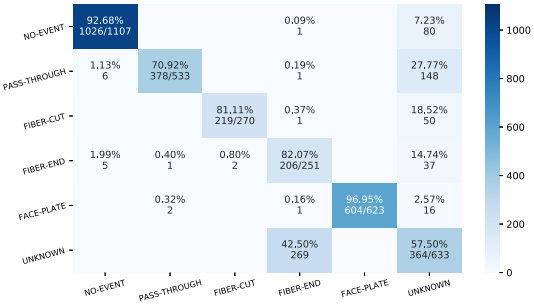
For $K = 1$, we analyze the condition in which the average peak of the heartbeats matches that of the OTDR events. With a closed-set classifier, all heartbeats are classified as FIBER-END (see Figure 4.11b). Softmax thresholding identifies



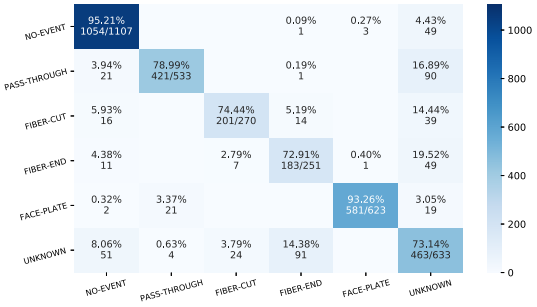
(a) Comparison between Softmax thresholding and OpenMax



(b) Base classifier



(c) Softmax thresholding



(d) OpenMax

Figure 4.11: We assess the *Event-Detection Network* on a test set of OTDR windows to which we have added ECG beat windows. Fig. a shows the True Positive Rate as the average intensity of the heartbeat changes. Figures b, c, and d show the confusion matrices in the closed-set case, where object classes are all known. In this case, we set a threshold on the Softmax probability and with OpenMax.

57.5% of the heartbeats as unknown, while 42.5% are classified as FIBER-END (see Figure 4.11c). OpenMax achieves the best performance, detecting 73.14% of the heartbeats as unknown events (as revealed in Figure 4.11d).

4.5.4 Known and Unknown Event Detection

In this section, we evaluate the performance of the *Event-Detection Network* in detecting both known and unknown events within the trace. First, in Sec. 4.5.4, we assess its performance on traces that contain only known events. Then, in Sec. 4.5.4, we introduce ECG beats in the traces, requiring the network to recognize them as unknown events. Furthermore, we provide a qualitative analysis of a newly introduced OTDR event, termed BULK-ATTENUATOR. This event represents an attenuation of the transmitted signal and must be reported, even if its label is not included in the training set.

	Accuracy		Precision		Recall		F_1 Score		ROC-AUC	
	Soft	Open	Soft	Open	Soft	Open	Soft	Open	Soft	Open
Pass-Through	0.681	**0.758**	0.616	**0.655**	0.681	**0.758**	0.647	**0.703**	0.790	**0.897**
Fiber-Cut	**0.156**	0.048	**0.123**	0.068	**0.155**	0.048	**0.137**	0.056	**0.487**	0.232
Fiber-End	0.661	**0.745**	0.641	**0.702**	0.661	**0.741**	0.651	**0.721**	0.855	**0.865**
Face-Plate	0.299	**0.737**	0.439	**0.727**	0.299	**0.737**	0.355	**0.732**	0.594	**0.829**
Mean	0.449	**0.572**	0.455	**0.538**	0.4492	**0.571**	0.448	**0.553**	0.681	**0.706**

Table 4.2: Leave-One-Class-Out classification results for OTDR events. We remove an event type at a time from the dataset (the one specified in each row), and we train the network based on the remaining data. Then we test using also instances of the left-out event, which must be labelled as unknown. For each metric, we compare Softmax and OpenMax.

	TE	TE+ECG		
	DetNet	DetNet	DetNet+OMDH	DetNet+Pre
Pass-Through	70.49	61.03	64.41	**64.43**
Fiber-Cut	73.87	56.87	63.40	**70.28**
Fiber-End	62.34	20.34	24.19	**55.64**
Face-Plate	70.39	63.39	67.01	**71.62**
Unknown (ECG)	–	0.0	6.20	**74.92**
mAP@.5	69.27	40.32	45.04	**67.38**
mAP@[.5:.05:.95]	38.86	21.49	26.71	**37.94**

Table 4.3: Detection results in terms of AP@0.5 from two test scenarios: one consisting solely of known OTDR events (TE) and another in which ECG tracings have been added (TE + ECG). We show the performance of the Event-Detection Network (DetNet). Then we compare its performance against the OpenMax embedded in the detection head (ODH) and the network with our preprocessing strategy (DetNet + Pre).

For the detection of unknown events, we use the OpenMax method with $\tau = 0.35$ and $\delta = 25$. It is important to note that the Leave-One-Class-Out strategy used to evaluate classification cannot be applied to detection experiments. Indeed, we would need to exclude all OTDR traces that contain even a

Event Type	NCS1001	DetNet
Reflective	22.43	**68.93**
Non-Reflective	68.08	**70.03**
End of Fiber	44.08	**68.29**
mAP	44.86	**69.08**

Table 4.4: Detection results in terms of AP@0.5. Comparison between the NCS-1001 and our Event-Detection Network.

single instance of the excluded event. Since each trace typically includes two to three event classes, this would significantly reduce the dataset, requiring the collection of a much larger dataset to train the network.

Known Event Detection

For the detection of known events, our *Event-Detection Network* achieves a mAP@0.5 of 69.27% (PASCAL VOC metric) across all classes and a mAP@[.5:.05:.95] of 38.86% (COCO metric), as shown in Table 4.3. We also compared our approach to the existing solutions currently deployed in Cisco NCS1001 [23] for OTDR event detection. To ensure a fair comparison, we mapped the predicted event types from our model to the standard categories used by the existing systems, which include fewer event types than our OTDR detection network. The main events are REFLECTIVE, NON-REFLECTIVE, and FIBER-END. As presented in Table 4.4, the results clearly show that our proposed detection network significantly outperforms the hand-crafted detectors currently implemented on NCS1001 devices. These detectors rely on thresholds fine-tuned by optical experts and function based on strict assumptions regarding the position and size of events. In contrast, our *Event-Detection Network* offers a broader set of event labels and more precise localization, as it avoids processing traces using fixed-size windows.

Unknown Event Detection

To evaluate the performance of our *Event-Detection Network* in identifying unknown events, we injected ECG tracings within OTDR traces. First, we tested our Event-Detection Network using a dataset with ECG tracings (TE + ECG). We

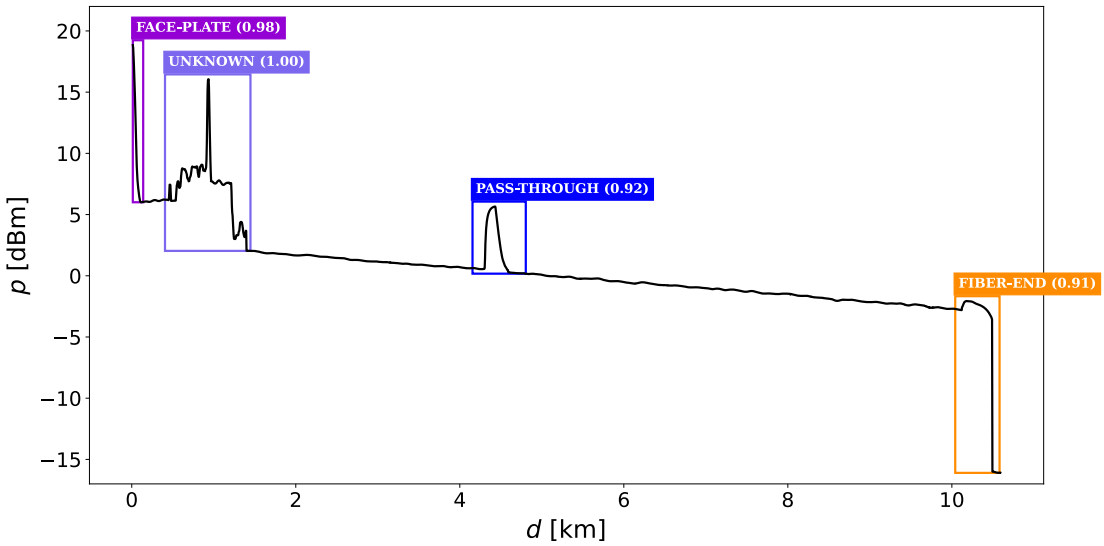


Figure 4.12: During training, *Bulk-Attenuator* instances were not provided, so the network either ignores or incorrectly classifies these events. The OTDR trace depicted shows that our algorithm can detect the *Bulk-Attenuator* instances as unknown events in addition to the known events.

obtained a mAP@0.5 of 40.32%, indicating a decrease of more than 25% compared to the scenario without ECG tracings. This result highlights the instability of a closed-set solution in real-world settings where new events can emerge over time. Then, we compared the Event-Detection Network (*DetNet*) equipped with OpenMax in the detection head (*DetNet+OMDH*) against our preprocessing strategy (*DetNet+Pre*). Our experiments demonstrated that training and testing the model on the original traces (denoted as T) yielded better event detection performance than using detrended traces (T'). This is likely because the trend retains essential information for event detection. Certain events tend to occur at specific locations within the trace. For instance, the FACE-PLATE typically appears at the beginning, while events such as FIBER-END and FIBER-CUT are usually found at the end. The network can leverage this positional information when analyzing T, but these cues are lost in T'.

DetNet+OMDH performed similarly to the base *DetNet*, but when tested on the dataset including ECG tracings, *DetNet+OMDH* achieved a mAP@0.5 of 45.04%, representing only a marginal 1.0% improvement over the base model. This slight enhancement is due to the missing region proposals for unknown events, which restricts the effectiveness of OpenMax. However, it is important to note that OpenMax in the detection head helps reduce false positives by classifying them as

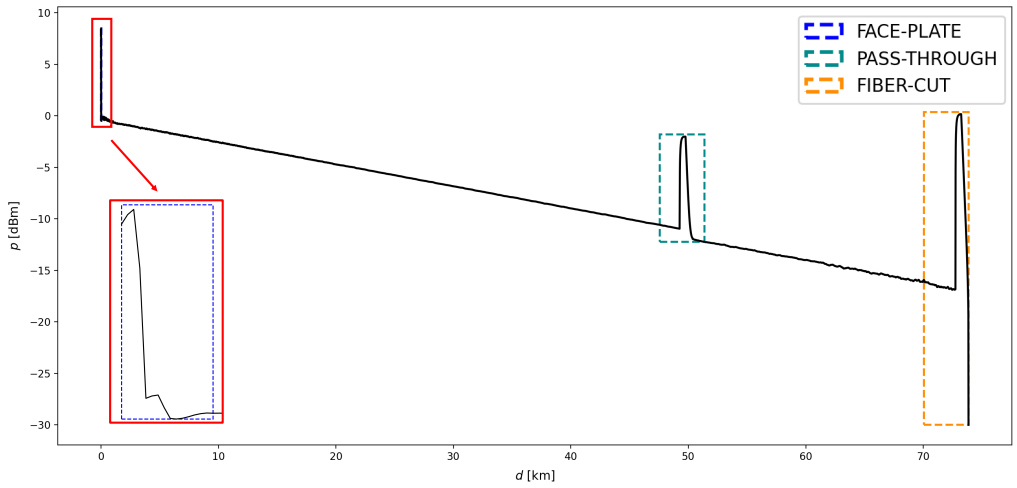


Figure 4.13: Qualitative results on a NCS1001 trace. Red boxes indicate the zoomed areas. All events have been correctly detected.

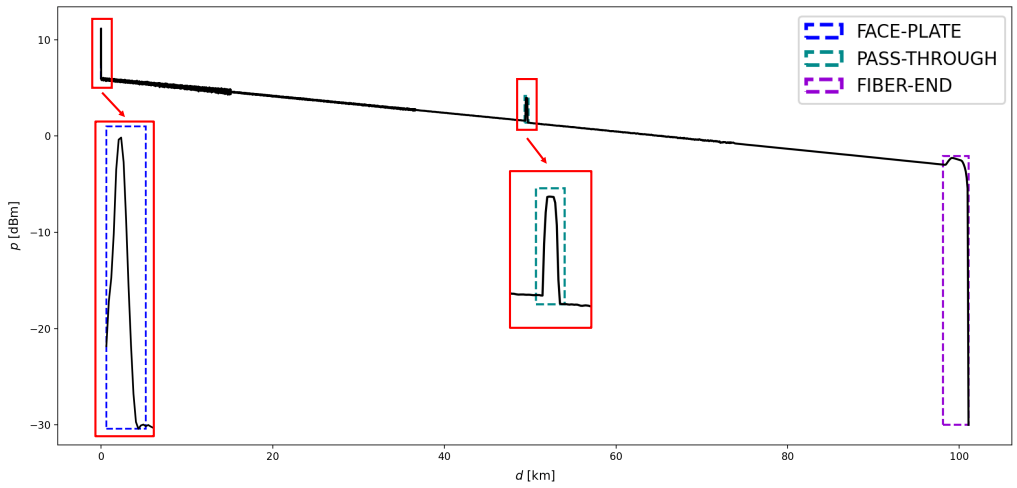


Figure 4.14: Qualitative results on a NCS1010 trace. Red boxes indicate the zoomed areas. All events have been correctly detected.

unknown. Our proposed solution, *DetNet+Pre*, detailed in Algorithm 2, achieved a mAP@0.5 of 68.12% on known OTDR events and 67.38% on the dataset with ECG tracings. We further assessed the model's performance in detecting unknown events using OTDR traces from *Cisco* customers, which included a previously unseen event, the BULK-ATTENUATOR. As shown in Fig. 4.12, our model successfully detected both the unknown and known events.

	NCS1001 (Domain $\mathcal{D}^{\mathcal{S}}$)		NCS1010 (Domain $\mathcal{D}^{\mathcal{T}}$)	
	DetNet	DetSys	DetNet	DetSys
Accuracy	92.45	**93.78**	57.91	**93.18**
Precision	77.91	**83.89**	13.77	**79.52**
Recall	**89.93**	88.32	14.67	**88.59**
F_1 Score	83.49	**86.05**	14.20	**83.81**
False Positive Rate	6.86	**4.69**	28.61	**5.67**
False Negative Rate	**10.06**	11.67	85.33	**11.40**
mAP@0.5	69.27	**75.33**	3.12	**64.93**
mAP@[.5 : .05 : .95]	38.86	**74.46**	0.83	**64.36**

Table 4.5: Comparison of detection performance between the Event-Detection Network (DetNet) and the Event-Detection System (DetSys) conducted using traces acquired from the NCS1001 and NCS1010 OTDR devices. The values for the NCS1001 were computed using cross-validation.

4.5.5 Event Detection under Domain Shift

In this section, we evaluate the performance of the *Event-Detection System* in detecting events across two types of devices, the NCS1001 and NCS1010. First, in Sec. 4.5.5, we assess the performance of the Interval Proposal Algorithm (IPA). Then, in Sec. 4.5.5, we test the overall detection system. Finally, we inspect the confusion matrices in Sec. 4.5.5.

Fig. 4.13 and Fig. 4.14 present examples of qualitative experiments on NCS1001 and NCS1010 traces, further demonstrating the accuracy of our *Event-Detection System* in detecting optical events within the trace.

Interval Proposal Algorithm

The IPA applied to NCS1001 traces achieves a recall of 96.45% and a precision of 67.1%. On NCS1010 traces, the results show a recall of 97.00%, though the precision drops to 31.19%. Although the precision is relatively low, the high recall is crucial as it indicates that nearly all events are captured and provided to the classifier, where many are likely to be classified as NO-EVENTS.

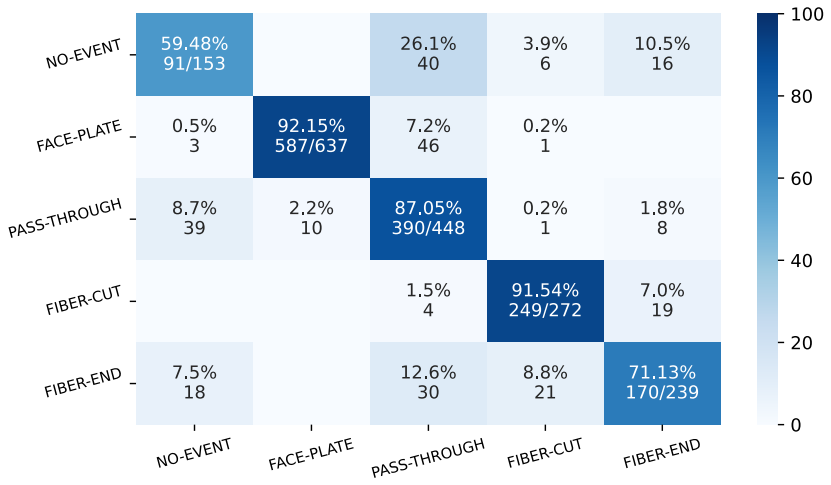


Figure 4.15: Confusion matrix computed by cross-validation on NCS1001 traces with our *Event-Detection System*. Rows denote the ground-truth, while columns denote the predictions.

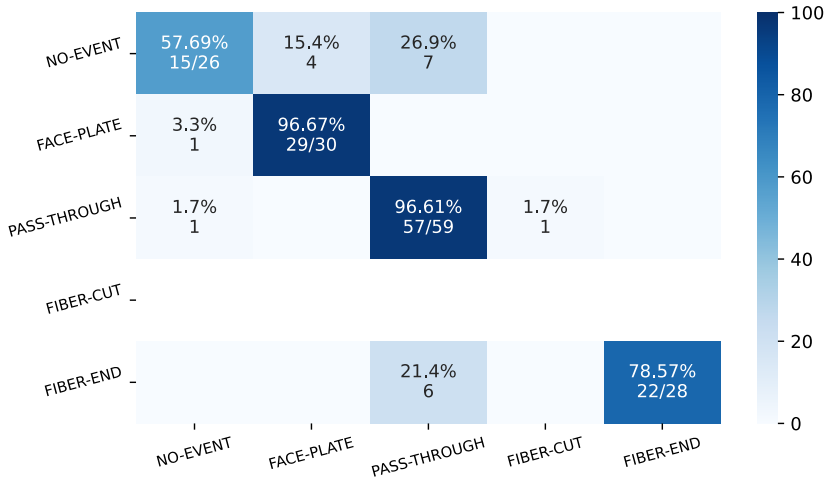


Figure 4.16: Confusion matrix computed from NCS1010 traces with our *Event-Detection System*. Rows denote the ground-truth, columns denote predictions.

Event-detection performance

The performance of our *Event-Detection System* is detailed in Table 4.5, where it is compared with the *Event-Detection Network* using traces from both the source domain, $\mathcal{D}^{\mathcal{S}}$ (NCS1001 device), and the target domain, $\mathcal{D}^{\mathcal{T}}$ (NCS1010 device). In the source domain $\mathcal{D}^{\mathcal{S}}$, the first two columns indicate that our method achieves

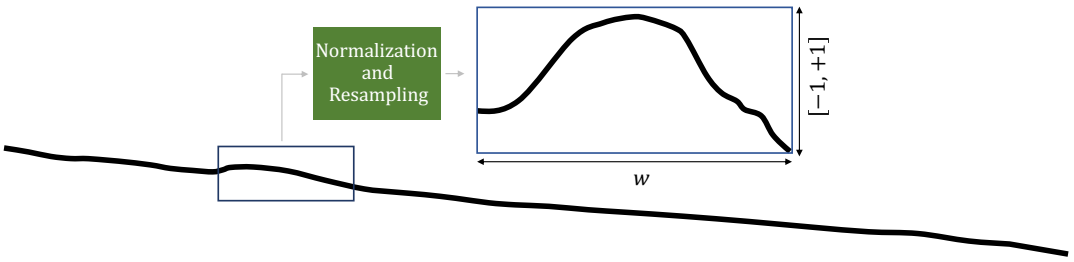


Figure 4.17: This negligible bump results in a PASS-THROUGH after the normalization.

results comparable to those of the *Event-Detection Network*. However, it is important to note that our method achieves a higher mean Average Precision (mAP). This improvement can be attributed to our expert-driven localization and interval estimation, which are more accurately aligned with the events. Additionally, the fact that increasing the Intersection-over-Union (IoU) threshold does not reduce the mAP further supports the robustness of our approach.

When tested on NCS1010 traces, the benefits of our *Event-Detection System* become more evident. The *Event-Detection Network* performs poorly, failing to detect most events. This is largely due to the differences in event shapes between the NCS1010 domain and the source domain $\mathcal{D}^{\mathcal{S}}$, as highlighted by the high false negative rate (FNR) and low mAP. In contrast, our method remains effective despite the domain shift, achieving performance similar to that of the source domain $\mathcal{D}^{\mathcal{S}}$.

Confusion matrices

To better inspect the errors of our method, we can inspect the confusion matrices in Fig. 4.15 and Fig. 4.16. These matrices highlight that the main source of error comes from misclassifying NO-EVENTS as PASS-THROUGHS, as indicated by the false positive rate of 11.4%. This occurs because small bumps in the trace can be incorrectly identified as PASS-THROUGHS. The normalization process contributes to this issue, making noisy NO-EVENTS resemble PASS-THROUGHS, as in Fig. 4.17.

4.6 Conclusions

We present two novel methods to address the detection of events in OTDR traces. In particular, we first describe an *Event-Detection Network* learned end-to-end that recognizes more event types than existing algorithms, but it is also more

accurate in localizing them. Remarkably, we have presented a solution to report unknown events that do not appear in the training set. Then we address Domain Adaptation in OTDR traces. Unlike the *Event-Detection System*, our *Event-Detection Network*, leverages domain knowledge to analyze traces from different domains, enabling the same model to perform event detection across various types of devices.

Our experiments show that the proposed approaches effectively solve the detection of optical events and can be used in a real-world environment, being employed in an embedded device and running in comparable time with respect to current industrial solutions (less than 6 seconds).

Chapter 5

Anomaly Detection in Optical Spectra

In DWDM systems, optical channels carry user data, forming an optical signal known as the *spectrum*. At the lower end of this spectrum, there is a noise contribution known as *Amplified Spontaneous Emission* (ASE), which is generated by *Erbium-Doped Fiber Amplifiers* (EDFAs) used to compensate for signal attenuation across fiber spans. Faults in the system can cause channels to deviate from their target power level, leading to *anomalies* that can disrupt data transmission. When the spectrum is transmitted over long distances, amplifiers introduce fluctuations and tilts, causing channels and ASE to follow a non-horizontal trend. Therefore, we define anomalies as peaks that deviate from the channel trend.

Although anomaly detection is an active research field [16, 29], it has not been extensively investigated in the context of optical spectra. In industrial applications, a common approach involves establishing two thresholds, T_L and T_H, on channel power, and identifying peaks as anomalies if they fall outside these boundaries. However, the two-threshold approach is ineffective when the channel trend is non-horizontal. Anomalies could be, in principle, detected by considering the spectrum as a time series and using a supervised detection network to identify channels and anomalies. This strategy adapts visual recognition models, such as the Faster R-CNN [80] and YOLO [78], to 1D signals. Recent studies have successfully applied this technique to detect anomalies in various contexts, including ECG tracings [101], seismic waves from earthquakes [98], and speech objects within fixed-length audio signals [86]. However, while localizing channels and anomalies is relatively straightforward, classifying between them is difficult since anomalies are rare and have the same shape as channels. The only difference is that their power levels deviate from the channel trend.

Our method estimates the channel trend and detects anomalies as candidate channels that deviate from it. To achieve this, we propose a joint optimization procedure that captures the relationship between the channel and ASE trends. Based on the observation that distortions consistently affect channel and ASE samples, we assume the trends must be similar, even in regions without samples.

Furthermore, to estimate the principal trends in the spectrum, we employ robust fitting techniques like RanSaC [33], which iteratively fit analytical models to randomly selected samples. RanSaC evaluates these models by measuring their consensus, namely the number of samples within a specified tolerance, referred to as the *inlier threshold*. The model with the highest consensus is then chosen.

5.1 Problem Formulation

An optical spectrum S can be represented as a collection of pairs:

$$S = \{(x_1, S(x_1)), (x_2, S(x_2)), \dots, (x_N, S(x_N))\} \tag{5.1}$$

where x_i denotes the frequency, and $S(x_i)$ indicates the power of the i-th sample, capturing the distribution of power across frequencies. Within this spectrum, there is an unknown number of channels, each occurring at arbitrary locations. Each channel is characterized by a triplet (c_j, b_j, p_j), where c_j represents the central frequency, b_j denotes the bandwidth, and p_j indicates the power level. At the lower end of the spectrum, we observe a baseline power level known as the Amplified Spontaneous Emission (ASE), which reflects the noise present in the transmission. Both the channels and the ASE samples exhibit unknown trends, denoted as ν and μ, respectively. We assume that these trends can be approximated by polynomial functions of degree d, allowing us to model their behavior across the C-band frequency range.

Our goal is to detect anomalies by identifying channels that deviate from the expected trend ν and estimating their central frequency c_k without supervision. This process is crucial for recognizing unexpected behaviors in the spectrum, which could indicate potential system issues or disruptions.

5.2 Methodology

Since channels and ASE undergo the same distortions during transmission, their trends must be similar. Our method captures this similarity through our joint optimization procedure, yielding a robust channel trend estimate, even with anomalies and limited channels. At a high-level, it consists of five stages:

Algorithm 3 Anomaly detection in optical spectra

Input: Spectrum S, inlier threshold ε
Output: Central frequencies c_k of the K anomalies

Stage 1: Robust trend fitting
Fit a linear trend $\ell \leftarrow$ LO-RanSaC$(x_i, S(x_i), \varepsilon)$
Stage 2: Channel and ASE clustering
$h_r \leftarrow$ Construct the histogram of residuals.
$\mathcal{C}, \mathcal{N} \leftarrow$ Otsu(h_r).
$\mathcal{C} \leftarrow$ FindPeaks$(\mathcal{C})$.
Stage 3: Initial guesses estimation
$\nu \leftarrow$ Fit polynomial to the samples in $\mathcal{C}$.
$\mu \leftarrow$ Fit polynomial to the samples in $\mathcal{N}$.
Stage 4: Joint optimization
$\nu^*, \mu^* \leftarrow$ Jointly estimate trends as in Eq. (5.5).
Stage 5: Anomaly detection
$c_k \leftarrow$ Recognize as anomalies $\{c_k \in \mathcal{C} : \text{err}(p_k, \nu) > \varepsilon\}$.

1. Robust trend fitting, where a linear trend is robustly fitted to the optical spectrum to determine the ASE tilt.

2. Channel and ASE clustering, which involves constructing a histogram of the residuals from the trend fitting, applying the Otsu method to cluster the data into candidate channels $\mathcal{C}$ and noise $\mathcal{N}$, and identifying peaks in the channel clusters.

3. Initial guesses estimation, where the channels and ASE trends, ν and μ, are independently fit on the channel and noise clusters, $\mathcal{C}$ and $\mathcal{N}$.

4. Joint optimization, which refine ν and μ to promote their similarity while maintaining data fidelity.

5. Anomaly detection, where channels that deviate from the previously estimated channel trend ν are recognized as anomalies.

These stages are further detailed below. A detailed description of the method is reported in Alg. 3.

Algorithm 4 Robust Linear Trend fitting by Sequential Lo-MSAC

Input: trace S, $\epsilon = \pm 0.5$ dBm tolerance, $k_{\max} = 100$ max iteration
Output: $\ell_1, \ldots, \ell_k$ linear trends

1: $X = S$
2: **while** $X \neq \emptyset$ **do**
3: $J^* = -\infty,\ k = 0$ ▷ Initialization
4: **while** $k < k_{\max}$ **do** ▷ Estimate linear trend
5: Select randomly two samples from S
6: Compute the line ℓ_k passing by the two points
7: Evaluate $J(\ell_k)$, the score of the line ℓ_k as in (5.2)
8: **if** $J(\ell_k) > J^*$ **then** ▷ Update the best solution
9: Run optimization to fit ℓ_k over inliers
10: $\ell^* = \ell_k$
11: $J^* = J(\ell_k)$
12: **end if**
13: $k = k + 1$
14: **end while**
15: Remove the inliers from X
16: **end while**

5.2.1 Robust Trend Fitting

In the first stage, we identify the spectrum tilt by robustly fitting a linear trend ℓ (Alg. 3, line 1). To achieve this, we employ a modified version of LO-RanSaC [20], which is described in Alg. 4, applying a loose inlier threshold of 3.0 dB to accomodate the spectrum fluctuations.

LO-RanSaC follows a two-step hypothesize-then-verify procedure. The hypothesize step is iterative: at each iteration k, we select a pair of points $(x_i, S(x_i))$ and $\big(x_j, S(x_j)\big)$ (Alg. 4, line 5) to instantiate a line ℓ_k (Alg. 4, line 6). The verification steps consist of keeping track of the line ℓ^* that best explains the data (Alg. 4, lines 7 - 13) among all the lines sampled so far. Lines are assessed using the score $J(\cdot)$ (line 7), defined as:

$$J(\ell) = \sum_{x_i} \rho\Big((x_i, S(x_i)), \ell\Big) \tag{5.2}$$

where $\rho(\cdot, \cdot)$ is a robust function that measures the distance between a line ℓ and a sample $(x_i, S(x_i))$, taking advantage of a user-specified tolerance ϵ that distinguishes between inliers and outliers. Specifically, ρ is defined as follows:

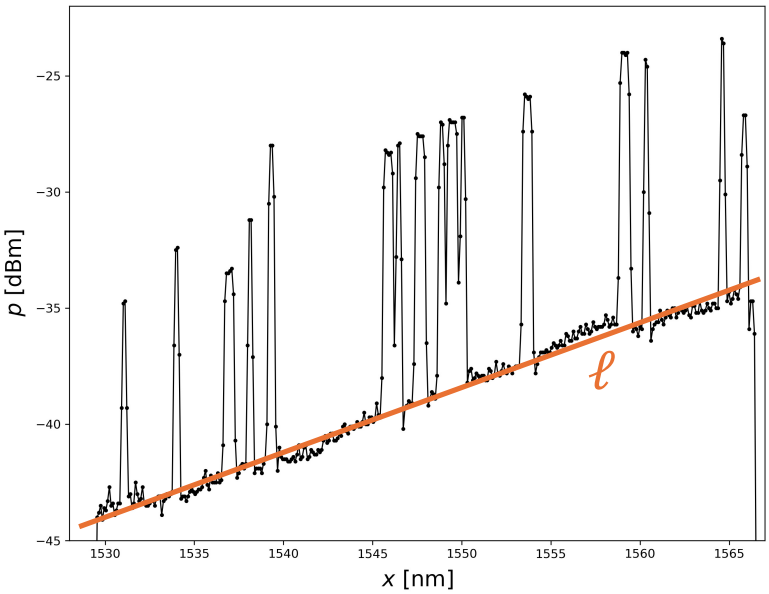


Figure 5.1: Optical spectrum where power is plotted against wavelength. The black points represent OCM samples, while the orange line represents the trend ℓ fitted using the RanSaC algorithm.

$$\rho\Big((x_i, S(x_i)), \ell\Big) = \begin{cases} d\Big((x_i, S(x_i)), \ell\Big) & \text{if } d\Big((x_i, S(x_i)), \ell\Big) < \epsilon \\ \epsilon & \text{otherwise.} \end{cases} \tag{5.3}$$

where, $d((x_i, S(x_i)), \ell) = |S(x_i) - mx_i - q|$ is the point-line distance along the vertical axis. In other words, when $(x_i, S(x_i))$ is an inlier for the line ℓ, $\text{err}(x_i, \ell)$ returns the distance between the line and the sample along the vertical direction. In contrast, when x_i lies outside the tolerance ϵ, a constant penalty ϵ is returned.

Since using only two points to fit a line lacks statistical efficiency, the returned line parameters may be very noisy. Therefore, an optimization that refines the parameters of the line on the inlier set is performed as soon as a better line is detected (Alg. 4, line 9). After iterating the process $m = 100$ times, the line ℓ^* with the minimum score is returned.

5.2.2 Channel and ASE Clustering

To coarsely cluster the channel and ASE samples, we compute the absolute residual d between each sample point and the fitted line ℓ^*. We generate a histogram from these residuals, as illustrated in Fig. 5.2.

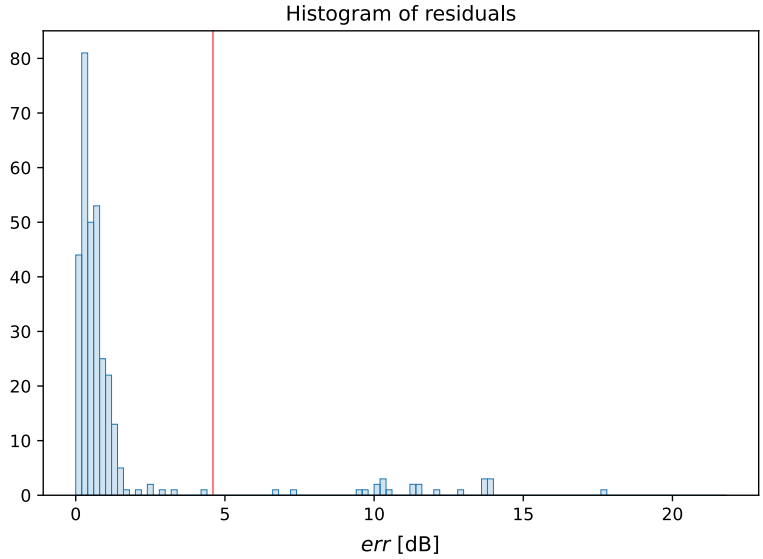


Figure 5.2: Histogram of the residuals between the observed data samples in S and the fitted line ℓ. Notice that the residuals are divided into two clusters: most are concentrated near zero, corresponding to noise, while the larger values correspond to detected peaks. The red vertical line represents the optimal threshold for separating noise from channels, computed using Otsu's method.

Then, we determine the optimal separation threshold (colored red in Fig. 5.2) between channel and ASE samples using the Otsu method [70] (see Alg. 3, line 3). This approach yields the optimal threshold that maximizes inter-class variance and minimizes intra-class variance. Samples with large residuals are classified as putative channels $\mathcal{C}$. In contrast, samples with small residuals, which align with the estimated ASE trend ℓ^*, are classified into the ASE cluster $\mathcal{N}$ (noise). The resulting clustering of channel and ASE samples is illustrated in Fig. 5.3.

Finally, within the cluster of putative channels $\mathcal{C}$, we identify local peaks, which serve as the most promising candidates for channels and anomalies (Alg. 3, line 4). The local peaks are marked by red dots in Fig. 5.4, highlighting the key areas of interest.

5.2.3 Initial guess estimation

We compute an initial guess for both the channel and ASE trends, ν and μ, by estimating the two polynomial trends independently on the two clusters $\mathcal{C}$ and $\mathcal{N}$. Specifically, we independently fit the channel trend $\nu(x) = \sum_{i=0}^{d} = m_i x^i$

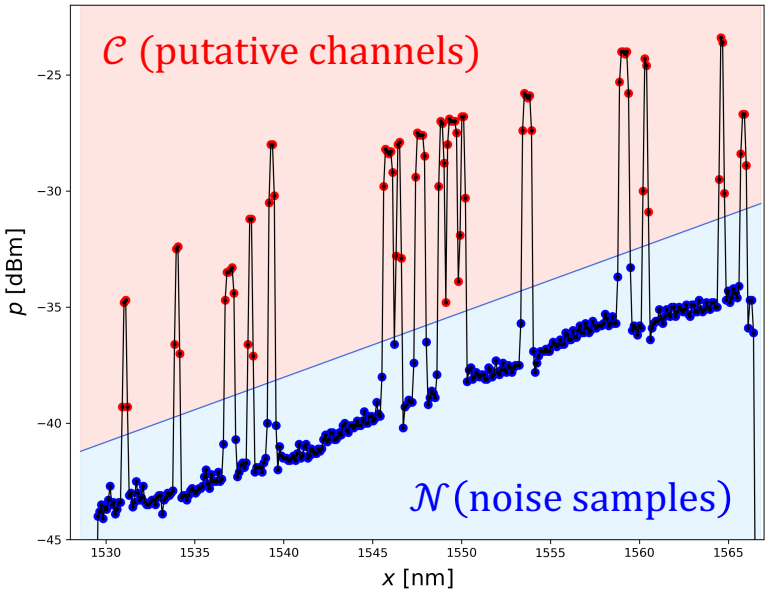


Figure 5.3: Dichotomization between the putative channels $\mathcal{C}$ (red region) and the background noise or ASE samples $\mathcal{N}$ (blue region). Red samples represent candidate channels, while blue samples represent noise.

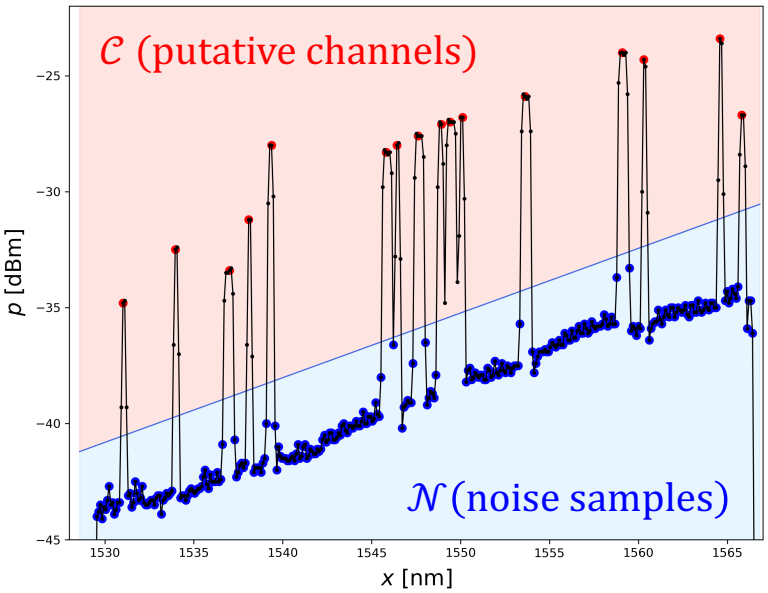


Figure 5.4: Identification of peaks in the spectrum. The candidate peaks (red points) are detected as local maxima in the set of putative channels $\mathcal{C}$ (red region).

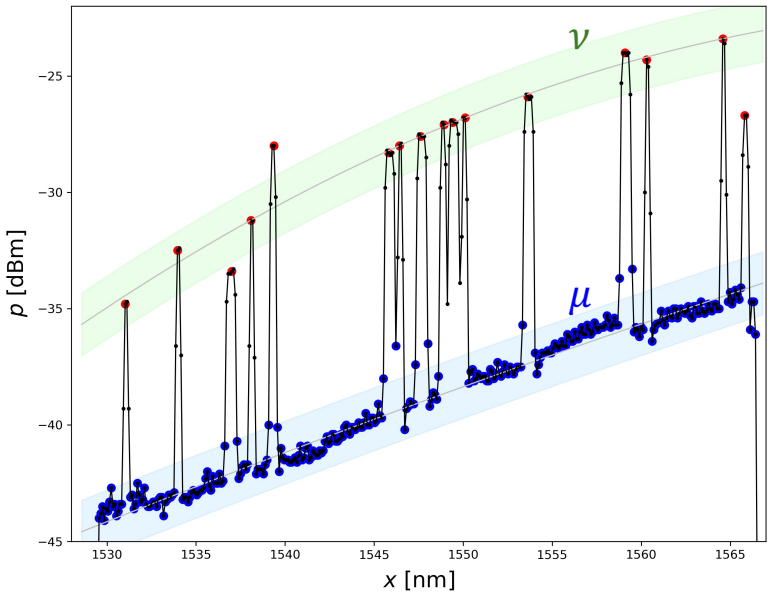


Figure 5.5: Estimation of the channel and ASE trends in the spectrum, denoted as ν^* (green) and μ^* (blue), respectively.

to the samples in $\mathcal{C}$ (Alg. 3, line 5) and the ASE trend $\mu(x) = \sum_{i=0}^{d} = n_i x^i$ to the samples in $\mathcal{N}$ (Alg. 3, line 6), where m_i and n_i are the coefficients of the polynomials. The fitting is performed by a LO-RanSaC [20] separately over the set of ASE and channels' clusters. The equations describing the loss of this fitting procedure is

$$\arg\min_{\mu,\nu} \sum_{p\in\mathcal{C}} \text{err}(p,\nu) + \sum_{p\in\mathcal{N}} \text{err}(p,\mu) \tag{5.4}$$

where $\text{err}(\cdot,\cdot)$ denotes the robust loss function defined in Eq. 5.3. The coefficients of these polynomials returned by the robust fitting are stored. Notice that when the spectra contain few channels, anomalies can influence the consensus, resulting in an utterly incorrect estimate. An example is shown in Fig. 5.10e, where the anomaly has been selected in the fitting due to the scarcity of channels. We further refine these initial guesses with our joint optimization procedure to avoid a wrong estimated trend. This prevents anomalies from influencing the estimated channel trend.

5.2.4 Joint optimization

Our method accounts for the similarity of the channel and ASE trends in a joint optimization procedure that provides a robust estimate of the channel trend ν,

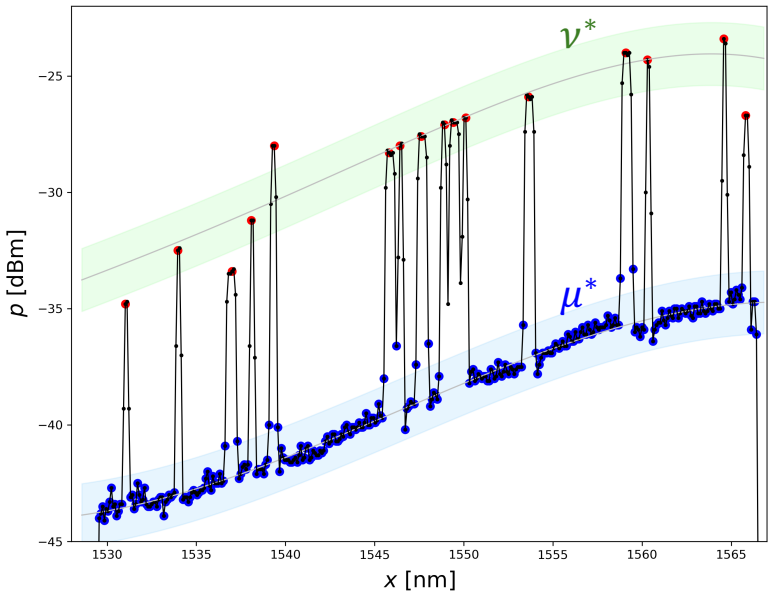


Figure 5.6: Result of the joint optimization procedure for correcting the spectrum trends of the channels and ASE. The two polynomial trends, ν^* (green) and μ^* (blue), represent the estimated trends for the channel and ASE, respectively. By leveraging the ASE trend, the channel trend effectively excludes the anomaly on the left, highlighting the optimization's success.

even in the presence of anomalies and in spectra having few channels.

The key insight is that the trend characterizing the channels should be close to the trend of the ASE, as both channels and ASE undergo the same amplification process. In contrast, anomalies due to wrong equalization or incomplete channel insertion will not follow the same trend.

We refine the initial guesses of ν and μ, by minimizing the following loss:

$$\arg\min_{\mu,\nu} \sum_{p\in\mathcal{C}} \mathrm{err}(p,\nu)^2 + \sum_{p\in\mathcal{N}} \mathrm{err}(p,\mu)^2 + \frac{\lambda}{d}\sum_{j=1}^{d}(n_j - m_j)^2 \tag{5.5}$$

where $\mathrm{err}(p,\nu)$ represents the distance (over the vertical axis) of the trend ν to samples $p \in \mathcal{C}$, and similarly $\mathrm{err}(p,\mu)$ represents the distance between samples $p \in \mathcal{N}$ and the trend μ. The first two terms in Eq. (5.5) measure the fidelity of the trends to the observed data, while the third term promotes similarity between the two trends, allowing ν and μ to align closely except for a vertical shift, represented by their zero-degree coefficients n_0 and m_0. The parameter $\frac{\lambda}{d}$ serves to balance the data fidelity and trend similarity terms, with the division by d

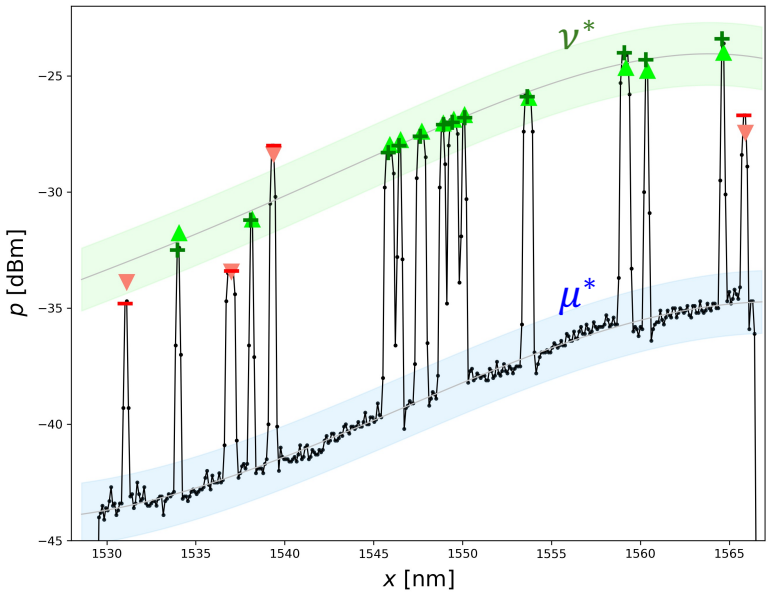


Figure 5.7: Anomalies are identified by deviations from the estimated channel trend, ν^* (green curve). Samples within the green-shaded region are considered channels, while those outside this region are outliers, marked as anomalies. The blue curve represents the ASE trend, μ^*, with samples within its shaded region classified as ASE samples. In this example spectrum, perfect detection is achieved. Green upward triangles and red downward triangles represent ground-truth channels and anomalies, while green plus signs and red minus signs indicate identified channels and detected anomalies.

enabling the use of a consistent value of λ across different trend orders d. It is important to note that the coefficients n_j and m_j are refined in each optimization step. We solve this non-linear optimization problem using the Levenberg-Marquardt [67] algorithm (Alg. 3, line 8). The results of our joint optimization procedure are illustrated in Fig. 5.6.

5.2.5 Anomaly Detection

Once the channel trend has been estimated by optimizing Eq. (5.5), detecting anomalies is straightforward. After having identified the channel trend ν, we recognize as anomalies the peaks $p \in \mathcal{C}$ that fall outside the inlier threshold ε of the channel trend, determining the green band illustrated in Fig. 1.5d. This is illustrated in Fig. 5.7.

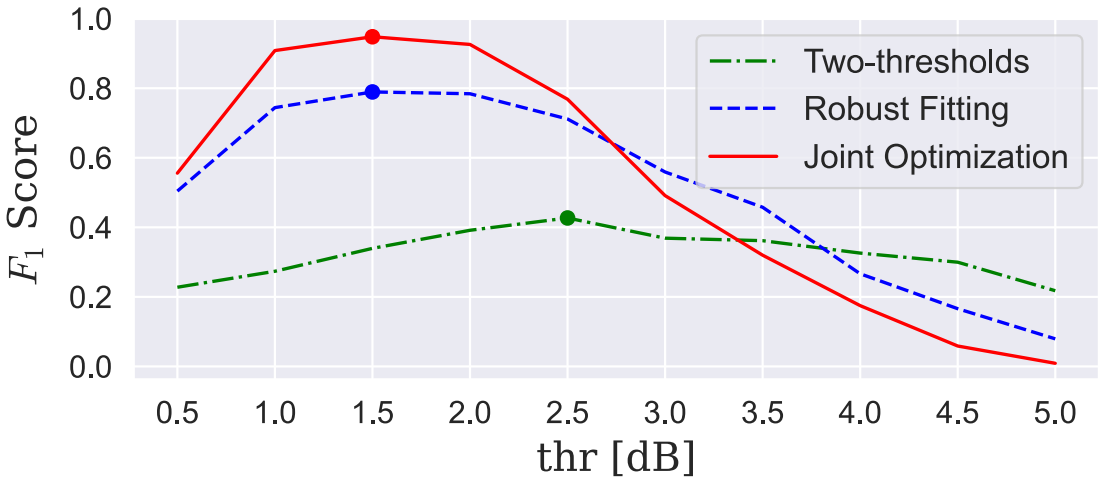


Figure 5.8: F_1 Score vs. inlier thresholds. For each method, we selected the threshold value leading to the highest F1 Score. The two-threshold approach achieves a maximum F_1 Score of 0.42 at 2.5 dB. The robust fitting approach and our joint optimization use a 1.5 dB threshold, achieving F_1 Scores of 0.79 and 0.95, respectively.

5.3 Experiments

This section provides both qualitative and quantitative evaluations of the proposed method. Experiments show that our joint optimization procedure significantly improves channel trend estimation, leading to more accurate anomaly detection. Additionally, while traditional robust fitting techniques may struggle with anomalies in spectra containing a limited number of channels, our method successfully addresses this issue.

5.3.1 Considered Methods

Two-thresholds: This approach is currently used by *Cisco* to detect anomalies in their systems. In particular, anomalies are identified by estimating all the peaks in S, as these indicate potential candidates for both channels and anomalies. We employ the peak detection algorithm described in [85]. Peaks are classified as anomalies if they lie outside two user-defined thresholds, T_L and T_H. This process can also be framed as a single threshold T and an inlier threshold ε. Peaks that fall outside the range $[T - \varepsilon, T + \varepsilon]$ are deemed anomalies. We set T to the mean power of the identified channels. In our experiments, we

established the inlier threshold at 2.5 dB, a value that has proven to optimize detection performance, as evidenced by the F_1 score shown in Fig. 5.8. It is worth noting that this methodology is considered the standard for industrial applications. Although this method does not account for the trends in the spectrum, it is considered due to its widespread use in real-world applications.

Robust Fitting: Robust techniques effectively handle outliers, which are common in our data. In fact, channels act as outliers for the ASE trend, while ASE samples act as outliers for the channel trend, resulting in poor trend estimation. Robust fitting techniques allow us to estimate the underlying trends even in the presence of outliers. To determine the channel trend, we first identify peaks using the method described in [85], as these may represent potential channels and anomalies. Next, we apply the LO-RanSaC algorithm [20] to fit a linear trend ℓ, setting the inlier threshold to 1.5 dB. This threshold has been shown to optimize performance, as illustrated in Fig. 5.8. Anomalies are then identified as peaks that are outliers to the fitted linear trend. It is important to note that linear trends are less affected by noise and anomalies compared to higher-order polynomials that yield worse results.

Faster R-CNN: Object detection methods have proven highly effective in capturing the spatial extent of complex objects within images. Motivated by this success, we have developed a 1D Faster R-CNN [80] that detects channels and anomalies by learning discriminative features in an end-to-end manner. Since channels and anomalies can exhibit complex patterns and vary significantly in size, intensity, and aspect ratio, a data-driven approach is well-suited, as it simultaneously optimizes both localization and classification across multiple scales. The network has been trained in a supervised manner using an annotated dataset of optical spectra. The central frequency of the output detections is determined by computing the centers of the bounding boxes. The model is trained with a 10^{-3} learning rate, which decays by 0.5 every 15 epochs. The feature extractor's receptive field is set to 278, and the spectra were resampled to ensure that the bandwidth of both channels and anomalies remains below 71 samples. This approach allows the network to adequately cover a significant portion of the spectrum in its predictions. The network is trained on a dataset of 102 synthetic spectra, using horizontal shifts and offset additions as the only forms of data augmentation. To address class imbalance in the dataset, a weighted loss function was implemented.

Joint Optimization: This is our solution where we model the trends as polynomials of order $d = 4$. Higher-order polynomials would be prone to overfitting, resulting in slow algorithm convergence. Our solution uses an inlier threshold of 1.5, the value for which it achieves the best performance, as depicted in Fig. 5.8.

	Accuracy		Precision		Recall		F_1 Score	
	Mean	Var.	Mean	Var.	Mean	Var.	Mean	Var.
Two-thresholds	0.662	0.054	0.329	0.119	0.697	0.141	0.391	0.112
Robust Fitting	0.951	0.010	0.800	0.117	0.810	0.110	0.789	0.106
Faster R-CNN	0.857	0.006	0.000	0.000	0.000	0.000	0.000	0.000
Joint Optimization	**0.989**	0.002	**0.968**	0.028	**0.937**	0.036	**0.948**	0.030

Table 5.1: Quantitative results over the synthetic test set of optical spectra.

5.3.2 Considered Datasets

We considered a dataset of 95 real-world spectra collected by Cisco in the field, with an example illustrated in Fig. 5.9. Acquiring extensive real-world spectra presents challenges, as they may contain sensitive user information, and annotated anomalies are scarce since most spectra are obtained under normal conditions. To address this limitation and obtain quantitative results, we generated a synthetic dataset comprising 165 optical spectra. In developing this dataset, we meticulously modeled the realistic distortions encountered in fiber optics, such as fluctuations and tilts introduced by amplifiers. Each spectrum includes between 1 and 50 channels, with a maximum of 5 anomalies per spectrum. In total, the dataset encompasses 2800 channels and 200 anomalies.

5.3.3 Figures of Merit

To assess all the aforementioned methods, we employ standard performance measures, such as accuracy, precision, recall, and F_1 score. We first match each estimated anomaly to the corresponding ground-truth using the Hungarian algorithm [68] that estimates the function σ that maps each central frequency c_i to its nearest estimate $\hat{c}_{\sigma(i)}$. An anomaly is successfully localized when the distance between $\hat{c}_{\sigma(i)}$ and c_i falls within the channel bandwidth:

$$c_i - \frac{b_i}{2} \leq \hat{c}_{\hat{i}} \leq c_i + \frac{b_i}{2}. \tag{5.6}$$

Unmatched peaks in the ground-truth are marked as false negatives, whereas unmatched predictions are marked as false positives.

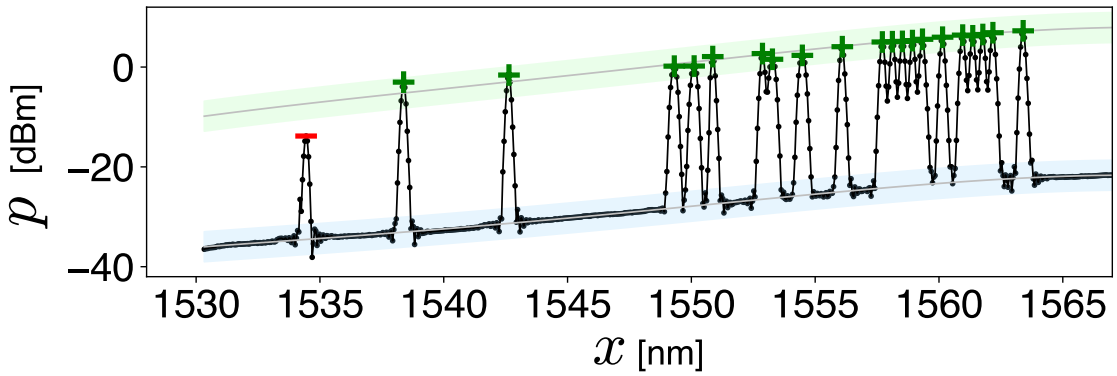


Figure 5.9: Real-world spectrum analyzed with our joint optimization.

5.3.4 Discussion

The threshold of each method (except the Faster R-CNN, which does not need one) has been determined by picking the values leading to the highest F_1 Score, as in Fig. 5.8. Results averaged over all the synthetic spectra are reported in Table 5.1 and indicate that our method is the best-performing algorithm according to all the selected metrics. Fig. 5.10 depicts three meaningful sample results.

The two-threshold approach achieves the best performance in spectrum S_1 having a nearly horizontal trend (Fig. 5.10a) but is ineffective on the tilted spectra S_2 and S_3 (Fig. 5.10b and Fig. 5.10c). Notice that the channels on the right-hand side of the spectrum are recognized as anomalies, as their peak power falls outside the green band. The robust fitting approach handles both the flat and tilted spectra S_1 and S_2 (Fig. 5.10d and Fig. 5.10e). Still, it is ineffective in the presence of few channels, as the case of spectrum S_3 (Fig. 5.10f) where it selected the anomaly to fit the channel trend, resulting in a completely incorrect estimate. The Faster R-CNN identifies all peaks in S_1, S_2, and S_3 as channels (Fig. 5.10g, Fig. 5.10h and Fig. 5.10i) due to the severe class imbalance in network training caused by the rarity of anomalies. However, it localizes all channels that act as true negatives in the accuracy computation. Indeed, accuracy reaches 85.7%, whereas precision, recall, and F_1 Score are zero.

Our joint optimization method successfully detects anomalies in the flat spectrum S_1 (Fig. 5.10j), as well as in the presence of heavy tilt (Fig. 5.10k) and few channels (Fig. 5.10l), represented by spectra S_1 and S_2. Leveraging the ASE trend in the optimization, our method achieves 98.9% accuracy and 94.8% F_1 Score. When applied to the real-world spectrum depicted in Fig. 5.9, which is highly tilted but presents almost no fluctuations, our method accurately detects the anomalies and estimates the channel trend.

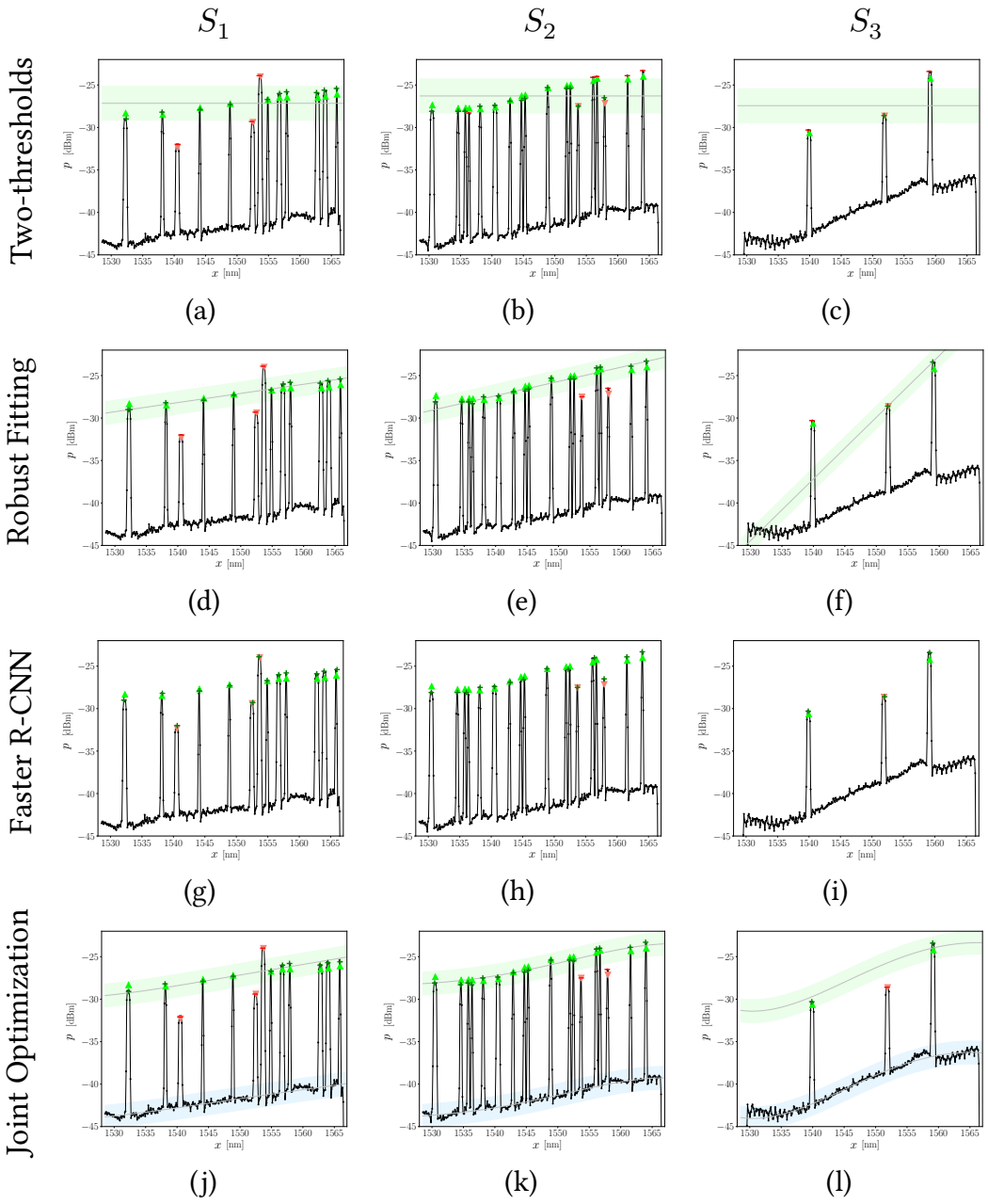


Figure 5.10: The three synthetic spectra S_1, S_2 and S_3 have been analysed with the different methods. True channels are denoted with the green triangles with the tip up, whereas true anomalies are denoted with the red triangle with the tip down. Estimated channels are denoted with the green ″+″, whereas estimated anomalies are denoted with the red ″—″.

5.4 Conclusions

We introduce a novel method for anomaly detection in optical spectra. The success of our approach lies in a joint optimization procedure that simultaneously fits the channel trend and extracts valuable information encoded in the ASE. Our solution operates in an unsupervised setting and effectively handles transmission distortions. Additionally, it is lightweight and well-suited for deployment on embedded devices.

Chapter 6

Concluding Remarks

This work presents learning-based methods that enable effective fault detection and monitoring while addressing domain shifts and limited supervision. First, we addressed monitoring in non-stationary environments, where samples are, in general, not independent and identically distributed (i.i.d.). Previous approaches primarily assume i.i.d. data, leading to poor performance in real-world monitoring tasks, such as fiber-optic networks. To address this problem, we presented a framework, *Hierarchical Change and Anomaly Detection* (HiCAD), capable of distinguishing between changes that are part of the in-control state and changes to an out-of-control state that likely indicates a fault. HiCAD is a general framework that can be applied to various domains, including healthcare. For example, in epilepsy prediction systems [99], the in-control process corresponds to the brain's physiological activity, as recorded by electroencephalograms (EEGs). In normal conditions, EEG signals span multiple frequency bands, each serving distinct functions in a context-sensitive manner. However, abnormal neural activity emerges before a seizure, causing deviations in the EEG signal. By leveraging our hierarchical framework, we can detect these abnormal neural activity patterns in advance, filtering out noisy data variations that make traditional change detection methods ineffective. While HiCAD enables robust monitoring, due to its data-driven nature, it lacks control over the Average Run Length for false alarms (ARL_0). Through rigorous experimentation on real-world datasets of optical network performance metrics from *Cisco* and *Microsoft*, we demonstrated that our method effectively detects changes to an out-of-control state and filters out the ones part of the in-control state. As a result, we developed and submitted a patent in collaboration with *Cisco*, reflecting the potential impact of our work.

Second, we addressed the monitoring of OTDR traces. Previous methods, which relied on rule-based approaches, could not discriminate between the finer

categories of faults that can occur in the fiber. In contrast, our 1D *Event-Detection Network*, trained end-to-end, recognizes a larger number of event types than existing algorithms and is more accurate in localizing them. In addition, our solution also reports unknown events, including rare ones not appearing in the training set. Our experiments show that the proposed approach can effectively detect optical events and be deployed in real-world environments, operating on an embedded device with processing times comparable to current industrial solutions (less than 6 seconds). To effectively address domain shifts across different OTDR devices, we proposed an *Event-Detection System* that decouples localization and classification, using signal processing techniques for event localization and a data-driven model for event classification. In particular, we exploit context features, which capture domain-specific information, and morphological features, which focus on the shape of the events, disregarding context-specific details. This approach enables our method to accurately predict events across different devices. Our method's current limitation is the need to configure the Interval Proposal Algorithm (IPA) for each OTDR device, which is responsible for the localization stage. This requirement is far less expensive than collecting entirely new datasets for retraining the model for a new device, as only a few traces are needed to fine-tune the IPA hyperparameters. Our approach can be applied to other domains beyond fiber-optic monitoring, where localizing and classifying specific shapes in the signal is necessary. For instance, in structural health monitoring [64], acoustic emission signals are employed to localize and classify crack locations. Our approach separates localization from classification, offering a key advantage: it enables the standardization of crack shapes across various materials. This is crucial because cracks appear differently depending on the material. By decoupling these tasks, we can standardize the representation of crack shapes, regardless of the underlying material properties.

Lastly, we addressed the monitoring of optical spectra using Optical Channel Monitoring (OCM) modules. Previous OCM methods rely on fixed thresholds for detecting anomalies, which fail at the receiver side due to spectrum distortions. Indeed, these methods do not account for the trends induced by the optical components, further limiting their accuracy. Our proposed approach exploits a joint optimization procedure to robustly estimate the channel trend and detect anomalies as deviations from it. This identifies potential faults and allows for timely interventions. Our approach addresses these challenges by assuming that fiber distortions can be modeled with a polynomial trend. We successfully capture deformations in most scenarios using a polynomial of degree 4 or 5. However, in complex systems, a higher-order polynomial may be required. While this improves flexibility, the increased number of parameters slows the algorithm's

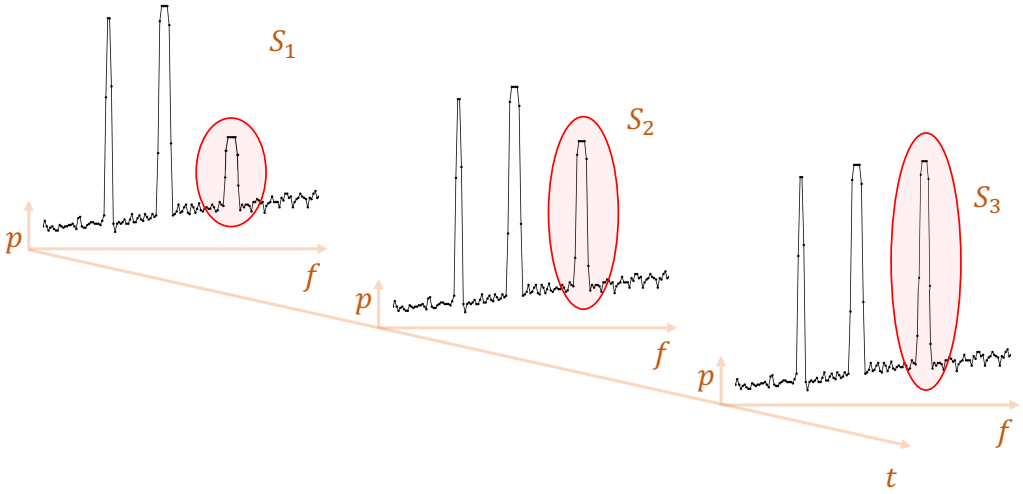


Figure 6.1: Example of the temporal evolution of a spectrum. A new channel (circled in red) is emerging.

convergence, which could pose challenges in real-time systems. The proposed approach has been patented in collaboration with *Cisco* and introduced in the NCS1001 [22] system.

Overall, the proposed methodologies are designed to generalize across different optical devices and operating conditions, enabling effective fault detection in fiber-optic networks without requiring extensive labeled datasets.

6.1 Research Directions

6.1.1 Time-varying Optical Spectra

In the context of optical channel monitoring, we can extend our method to consider multiple spectra simultaneously, incorporating the temporal dimension and allowing us to analyze the spectrum's temporal evolution. Indeed, by analyzing the transmission history, we can identify the causes of anomalies and trigger appropriate countermeasures.

The evolution of the spectrum over time provides additional information that can be profitably analyzed to better decide, in the presence of anomalies, when and which corrections to apply. When multiple OCM scans of the same spectrum are acquired at different times t, the trend of the ASE $\mu(x, t)$ and of channels

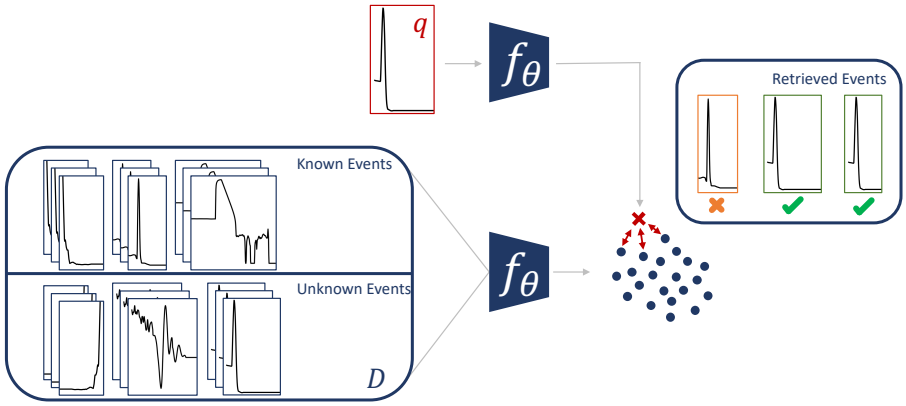


Figure 6.2: Scheme for unknown event retrieval. Given a dataset D of known and unknown events, we compute their latent representation using a function f_θ. At test time, we compute the latent representation of the query event q, and we compute the k Nearest Neighbors in that space. In the example, two out of three neighbors belong to the same ground-truth class of the query event q.

$\nu(x,t)$ exhibit temporal consistency. An example is shown in Fig. 6.1, where a channel (circled in red) is emerging. A possible formalization of this scenario consists in describing the sequence of spectra as a high dimensional time series $\{S_1, S_2, \dots, S_T\}$ of length T, where each spectrum corresponds to a point. We do not expect OCM recordings in this time series to be exact replicas of each other. Still, there will be rather intrinsic fluctuations of the spectra over time, which prevents analysis based on straightforward comparisons.

For instance, while persistent anomalies might hinder transmission and necessitate human intervention, transient anomalies might indicate channels introduced during spectrum acquisition that do not require intervention.

To successfully monitor these time series, we need to *i)* model and learn the temporal evolution of OCM spectra; *ii)* estimate channel locations and obtain anomaly information; *iii)* adopt high-dimensional sequential monitoring techniques for detecting changes. Analyzing OCM time series can help identify the causes of each anomaly and determine whether they are due to starting/ending channels or to a system problem (e.g., cables not properly connected). Moreover, we can detect frequency drifts and determine whether these depend on the limited accuracy of the scanning device or to some fault.

6.1.2 Similarity Search for Optical Events

When analyzing OTDR traces, the ability to detect unknown events allows us to identify faults that do not fit the pre-defined categories. However, exploring similarities among these unknown events can uncover new fault classes, gain insights into potential root causes, and enable domain experts to analyze them. Formally, we assume an input query event e_q and a collection D of known and unknown events. We aim to retrieve the k most similar events to e_q from D. This presents a challenge since unknown events in D are a minority, making it unsuitable for training an ad-hoc model.

To tackle this issue, we can employ an encoder f_θ to project the query event e_q into a latent space. The Euclidean distances within this space serve as a dissimilarity measure, enabling us to identify the k nearest neighbors to the embedding of e_q. This methodology is illustrated in Fig. 6.2. We can use the feature extractor from the event classifier as encoder f_θ. We expect the features learned for distinguishing known events can also be used for unknown ones. Alternatively, the encoder can be trained from scratch in a self-supervised manner, using both known and unknown events.

Bibliography

[1] Govind Agrawal. *Fiber-Optic Communication Systems*. Wiley India Pvt. Limited, 2007. ISBN: 9788126513864. URL: https://books.google.it/books?id=lyXQWQJYFrUC.

[2] N. Ahmed, T. Natarajan, and K.R. Rao. "Discrete Cosine Transform." In: *IEEE Transactions on Computers* C-23.1 (1974), pp. 90–93. DOI: 10.1109/T-C.1974.223784.

[3] Metin Aktas et al. "Deep learning based multi-threat classification for phase-OTDR fiber optic distributed acoustic sensing applications." In: *Fiber Optic Sensors and Applications XIV*. Vol. 10208. International Society for Optics and Photonics. 2017, 102080G.

[4] Cesare Alippi, Giacomo Boracchi, and Manuel Roveri. "Hierarchical Change-Detection Tests." In: *IEEE Transactions on Neural Networks and Learning Systems* 28.2 (2017), pp. 246–258. DOI: 10.1109/TNNLS.2015.2512714.

[5] Michèle Basseville and Igor Nikiforov. *Detection of Abrupt Change Theory and Application*. Vol. 15. Prentice-Hall, Inc, Apr. 1993. ISBN: 0-13-126780-9.

[6] Abhijit Bendale and Terrance E Boult. "Towards open set deep networks." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 1563–1572.

[7] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. 1st ed. Section 6.3. Springer, 2007. ISBN: 0387310738.

[8] F. Boitier et al. "Proactive Fiber Damage Detection in Real-time Coherent Receiver." In: *2017 European Conference on Optical Communication (ECOC)*. 2017 European Conference on Optical Communication (ECOC). Gothenburg: IEEE, 2017, pp. 1–3. ISBN: 978-1-5386-5624-2. DOI: 10.1109/ECOC.2017.8346077. URL: https://ieeexplore.ieee.org/document/8346077/ (visited on 04/29/2024).

[9] Konstantinos Bousmalis et al. "Domain Separation Networks." In: *Advances in Neural Information Processing Systems 29*. Aug. 2016.

[10] Konstantinos Bousmalis et al. "Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks." In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 95–104.

[11] R.G. Brown. *Exponential Smoothing for Predicting Demand*. Little, 1956.

[12] Andrea Carena et al. "Modeling of the Impact of Nonlinear Propagation Effects in Uncompensated Optical Coherent Transmission Links." In: *Journal of Lightwave Technology* 30 (2012), pp. 1524–1539.

[13] A.B. Carlson and P.B. Crilly. *Communication Systems*. McGraw-Hill Higher Education, 2010. ISBN: 9780077417222. URL: https://books.google.it/books?id=8qOUCgAAQBAJ.

[14] Diego Carrera and Giacomo Boracchi. "Generating High-Dimensional Datastreams for Change Detection." In: *Big Data Research* 11 (Oct. 2017). DOI: 10.1016/j.bdr.2017.09.001.

[15] Diego Carrera et al. "Domain Adaptation for Online ECG Monitoring." In: *2017 IEEE International Conference on Data Mining (ICDM)*. 2017, pp. 775–780. DOI: 10.1109/ICDM.2017.91.

[16] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly Detection: A Survey." In: *ACM Comput. Surv.* 41.3 (July 2009). ISSN: 0360-0300. DOI: 10.1145/1541880.1541882. URL: https://doi.org/10.1145/1541880.1541882.

[17] Guangyao Chen et al. "Adversarial Reciprocal Points Learning for Open Set Recognition." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021), pp. 1–1. DOI: 10.1109/TPAMI.2021.3106743.

[18] Xiaoliang Chen et al. "Self-Taught Anomaly Detection With Hybrid Unsupervised/Supervised Machine Learning in Optical Networks." In: *Journal of Lightwave Technology* 37.7 (2019), pp. 1742–1749.

[19] Ondřej Chum, Jiří Matas, and Josef Kittler. "Locally Optimized RANSAC." In: *Pattern Recognition*. Ed. by Bernd Michaelis and Gerald Krell. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 236–243. ISBN: 978-3-540-45243-0.

[20] Ondřej Chum, Jiří Matas, and Josef Kittler. "Locally Optimized RANSAC." In: *Pattern Recognition*. Ed. by Bernd Michaelis and Gerald Krell. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 236–243.

[21] Cisco. *Cisco Network Convergence System (NCS) 1010 Data Sheet*. https://www.cisco.com/c/en/us/products/collateral/optical-networking/network-convergence-system-1000-series/network-conver-system-1010-ds.html. Accessed: 2024-09-21.

[22] Cisco. *Cisco Network Convergence System 1001 Data Sheet*. https://www.cisco.com/c/en/us/products/collateral/optical-networking/network-convergence-system-1000-series/datasheet-c78-738782.html. Accessed: 2022-01-28.

[23] Cisco. *Cisco Network Convergence System 1001 OTDR Line Card Data Sheet*. https://www.cisco.com/c/en/us/products/collateral/optical-networking/network-convergence-system-1000-series/datasheet-c78-742294.html. Accessed: 2022-01-28.

[24] Cisco. *Cisco Routed Optical Networking Solution Guide, Release 2.0*. https://www.cisco.com/c/en/us/td/docs/optical/ron/2-0/solution/guide/b-ron-solution-20/m-ron.html. Accessed: 2024-09-14.

[25] Cisco. *Pioneering the IP and Optical Transformation*. https://www.cisco.com/c/en/us/products/optical-networking/white-paper-sp-ip-optimized-optical-transport.html. Accessed: 2024-08-10.

[26] Cisco. *Pioneering the IP and Optical Transformation*. https://www.cisco.com/c/en/us/products/optical-networking/white-paper-sp-ip-optimized-optical-transport.html. Accessed: 2023-06-30.

[27] Bruno Crosignani. *Le fibre ottiche nelle telecomunicazioni / Bruno Crosignani, Giancarlo De Marchis*. ita. 2. ed. Roma: Siderea, 1988.

[28] Akshay Dhamija et al. "The overlooked elephant of object detection: Open set." In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 1021–1030.

[29] Rémi Domingues et al. "A comparative evaluation of outlier detection algorithms: experiments and analyses." In: *Pattern Recognition, Volume 74, February 2018* (2017).

[30] Mark Everingham et al. "The Pascal Visual Object Classes (VOC) Challenge." In: *International Journal of Computer Vision* 88.2 (2010), pp. 303–338. ISSN: 1573-1405. DOI: 10.1007/s11263-009-0275-4. URL: https://doi.org/10.1007/s11263-009-0275-4.

[31] Hassan Ismail Fawaz et al. "Deep learning for time series classification: a review." In: *Data Mining and Knowledge Discovery* 33.4 (2019), pp. 917–963.

[32] Hassan Ismail Fawaz et al. "Inceptiontime: Finding alexnet for time series classification." In: *Data Mining and Knowledge Discovery* 34.6 (2020), pp. 1936–1962.

[33] Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography." In: *Commun. ACM* 24.6 (1981), pp. 381–395.

[34] Sergio Focardi et al. *Fisica Generale - Elettromagnetismo*. seconda edizione. Casa Editrice Ambrosiana, 2021.

[35] Luca Frittoli, Diego Carrera, and Giacomo Boracchi. "Change Detection in Multivariate Datastreams Controlling False Alarms." In: *Machine Learning and Knowledge Discovery in Databases. Research Track*. Cham: Springer International Publishing, 2021, pp. 421–436. ISBN: 978-3-030-86486-6.

[36] Luca Frittoli, Diego Carrera, and Giacomo Boracchi. "Nonparametric and Online Change Detection in Multivariate Datastreams Using QuantTree." In: *IEEE Transactions on Knowledge and Data Engineering* (2022), pp. 1–14. ISSN: 2326-3865. DOI: 10.1109/tkde.2022.3201635. URL: http://dx.doi.org/10.1109/TKDE.2022.3201635.

[37] Yaroslav Ganin et al. "Domain-Adversarial Training of Neural Networks." In: *Journal of Machine Learning Research* 17.59 (2016), pp. 1–35. URL: http://jmlr.org/papers/v17/15-239.html.

[38] Gregory C. Reinsel George E. P. Box Gwilym M. Jenkins and Greta M. Ljung. *Time Series Analysis: Forecasting and Control*. John Wiley and Sons Inc., 1970.

[39] Ross Girshick. "Fast R-CNN." In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1440–1448.

[40] Ross Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 580–587.

[41] Ary L Goldberger et al. "PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals." In: *circulation* 101.23 (2000), e215–e220.

[42] 650 Grouè. *Routed Optical Networking Leadership Report*. https://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/routed-optical-networking/650-group-ron-leadership-report.pdf. Accessed: 2023-06-30.

[43] Kaiming He et al. "Deep residual learning for image recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[44] Ezra Ip et al. "Coherent detection in optical fiber systems." In: *Opt. Express* 16.2 (Jan. 2008), pp. 753–791. DOI: 10.1364/OE.16.000753. URL: https://opg.optica.org/oe/abstract.cfm?URI=oe-16-2-753.

[45] KJ Joseph et al. "Towards open world object detection." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 5830–5840.

[46] R. E. Kalman and R. S. Bucy. "New Results in Linear Filtering and Prediction Theory." In: *Journal of Basic Engineering* 83.1 (Mar. 1961), pp. 95–108. ISSN: 0021-9223. DOI: 10.1115/1.3658902. URL: https://doi.org/10.1115/1.3658902.

[47] Kathan Kashiparekh et al. "Convtimenet: A pre-trained deep convolutional neural network for time series classification." In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–8.

[48] Lars E. Kruse et al. "Experimental Validation of Machine Learning-Based Joint Failure Management and Quality of Transmission Estimation." In: *IEEE Photonics Journal* 15.6 (Dec. 2023), pp. 1–9. ISSN: 1943-0655, 1943-0647. DOI: 10.1109/JPHOT.2023.3333420. URL: https://ieeexplore.ieee.org/document/10319670/.

[49] Ludmila I. Kuncheva. "Change Detection in Streaming Multivariate Data Using Likelihood Detectors." In: *IEEE Transactions on Knowledge and Data Engineering* 25.5 (2013), pp. 1175–1180. DOI: 10.1109/TKDE.2011.226.

[50] Marc Lavielle and G. Teyssière. "Detection of multiple change-points in multivariate time series." In: *Lithuanian Mathematical Journal* 46 (July 2006), pp. 287–306. DOI: 10.1007/s10986-006-0028-9.

[51] Shuang Li et al. "M-Statistic for Kernel Change-Point Detection." In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015.

[52] Yanghao Li et al. "Adaptive Batch Normalization for practical domain adaptation." In: *Pattern Recognition* 80 (2018), pp. 109–117. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2018.03.005. URL: https://www.sciencedirect.com/science/article/pii/S003132031830092X.

[53] Yimeng Li and Jana Košecká. "Uncertainty Aware Proposal Segmentation for Unknown Object Detection." In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2022, pp. 241–250.

[54] Sascha Liehr et al. "Real-time dynamic strain sensing in optical fibers using artificial neural networks." In: *Optics express* 27.5 (2019), pp. 7405–7425.

[55] Min Lin, Qiang Chen, and Shuicheng Yan. "Network In Network." In: *2nd International Conference on Learning Representations, ICLR*. 2014.

[56] Tsung-Yi Lin et al. "Focal loss for dense object detection." In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[57] Tsung-Yi Lin et al. "Microsoft COCO: Common objects in context." In: *European Conference on Computer Vision*. Springer. 2014, pp. 740–755.

[58] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation Forest." In: *2008 Eighth IEEE International Conference on Data Mining*. 2008, pp. 413–422. DOI: 10.1109/ICDM.2008.17.

[59] Wei Liu et al. "SSD: Single shot multibox detector." In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[60] Jie Lu et al. "Learning under Concept Drift: A Review." In: *IEEE Transactions on Knowledge and Data Engineering* 31.12 (2019), pp. 2346–2363. DOI: 10.1109/TKDE.2018.2876857.

[61] Helmut Lütkepohl. *New Introduction to Multiple Time Series Analysis*. Springer, 2005.

[62] Barnoski M. K. et al. "Optical time domain reflectometer." In: *Applied Optics* 16.9 (1977), pp. 2375–2379. DOI: 10.1364/AO.16.002375. URL: http://opg.optica.org/ao/abstract.cfm?URI=ao-16-9-2375.

[63] Virendra N. Mahajan. *Fundamentals of geometrical optics*. SPIE, Jan. 2014, pp. 1–445. DOI: 10.1117/3.1002529.

[64] Jonathan Melchiorre et al. "Acoustic emission onset time detection for structural monitoring with U-Net neural network architecture." In: *Developments in the Built Environment* 18 (2024), p. 100449. ISSN: 2666-1659. DOI: https://doi.org/10.1016/j.dibe.2024.100449. URL: https://www.sciencedirect.com/science/article/pii/S2666165924001303.

[65] Microsoft Networking Research Group. *Wide-Area Optical Backbone Performance*. https://www.microsoft.com/en-us/research/project/microsofts-wide-area-optical-backbone/overview/. 2017.

[66] Dimity Miller et al. "Dropout sampling for robust object detection in open-set conditions." In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 3243–3249.

[67] Jorge J. Moré. "The Levenberg-Marquardt algorithm: Implementation and theory." In: *Numerical Analysis*. Ed. by G. A. Watson. Berlin, Heidelberg: Springer Berlin Heidelberg, 1978, pp. 105–116. ISBN: 978-3-540-35972-2.

[68] James Munkres. "Algorithms for the Assignment and Transportation Problems." In: *Journal of the Society for Industrial and Applied Mathematics* 5.1 (1957), pp. 32–38.

[69] *Optical Time Domain Reflectometer (OTDR) Data Format*. SR-4731. Telcordia Technologies. 2011.

[70] Nobuyuki Otsu. "A Threshold Selection Method from Gray-Level Histograms." In: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (1979), pp. 62–66. DOI: 10.1109/TSMC.1979.4310076.

[71] Poojan Oza and Vishal M Patel. "C2ae: Class conditioned auto-encoder for open-set recognition." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2307–2316.

[72] Dimitri Palaz, Gabriel Synnaeve, and Ronan Collobert. "Jointly Learning to Locate and Classify Words Using Convolutional Networks." In: *INTERSPEECH*. 2016, pp. 2741–2745.

[73] Xiaomeng Peng et al. "Unsupervised Adaptive Fleet Battery Pack Fault Detection With Concept Drift Under Evolving Environment." In: *IEEE Transactions on Automation Science and Engineering* 21.3 (2024), pp. 2276–2288. DOI: 10.1109/TASE.2024.3363002.

[74] Pierluigi Poggiolini. "The GN Model of Non-Linear Propagation in Uncompensated Coherent Optical Systems." In: *J. Lightwave Technol.* 30.24 (Dec. 2012), pp. 3857–3879. URL: https://opg.optica.org/jlt/abstract.cfm?URI=jlt-30-24-3857.

[75] S. Rajpoot et al. "Future Trends in Fiber Optics Communication." In: *International Journal on Cybernetics and Informatics* (Apr. 2017). DOI: 10.5121/ijci.2017.6203.

[76] Joseph Redmon and Ali Farhadi. "YOLO9000: better, faster, stronger." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7263–7271.

[77] Joseph Redmon et al. "You Only Look Once: Unified, real-time object detection." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 779–788.

[78] Joseph Redmon et al. "You Only Look Once: Unified, real-time object detection." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 779–788.

[79] Shaoqing Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[80] Shaoqing Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[81] Gordon Ross, Dimitris Tasoulis, and Niall Adams. "Nonparametric Monitoring of Data Streams for Changes in Location and Scale." In: *Technometrics* 53 (Jan. 2012), pp. 379–389. DOI: 10.1198/TECH.2011.10069.

[82] Kuniaki Saito et al. "Learning to Detect Every Thing in an Open World." In: *Computer Vision – ECCV 2022*. Ed. by Shai Avidan et al. Cham: Springer Nature Switzerland, 2022, pp. 268–284. ISBN: 978-3-031-20053-3.

[83] Walter J Scheirer et al. "Toward open set recognition." In: *IEEE transactions on pattern analysis and machine intelligence* 35.7 (2012), pp. 1757–1772.

[84] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. "Anomaly detection in time series: a comprehensive evaluation." In: *Proc. VLDB Endow.* 15.9 (May 2022), pp. 1779–1797. ISSN: 2150-8097. DOI: 10.14778/3538598.3538602. URL: https://doi.org/10.14778/3538598.3538602.

[85] Scipy. *Find peaks*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html. Accessed: 2022-10-10.

[86] Yael Segal, Tzeviya Sylvia Fuchs, and Joseph Keshet. "SpeechYOLO: Detection and Localization of Speech Objects." In: *Proc. Interspeech 2019* (2019), pp. 4210–4214.

[87] Yi Shi et al. "An event recognition method for Φ-OTDR sensing system based on deep learning." In: *Sensors* 19.15 (2019), p. 3421.

[88] Lihi Shiloh, Avishay Eyal, and Raja Giryes. "Deep learning approach for processing fiber-optic DAS seismic data." In: *Optical Fiber Sensors*. Optical Society of America. 2018, ThE22.

[89] Alexander Tartakovsky, Igor Nikiforov, and Michèle Basseville. *Sequential Analysis: Hypothesis Testing and Changepoint Detection*. Chapman & Hall, Aug. 2014. ISBN: 9781439838204. DOI: 10.1201/b17279.

[90] Ashish Vaswani et al. "Attention is All You Need." In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017.

[91] Sagar Vaze et al. "Open-Set Recognition: A Good Closed-Set Classifier is All You Need." In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=5hLP5JY9S2d.

[92] A. P. Vela et al. "Soft Failure Localization During Commissioning Testing and Lightpath Operation." In: *Journal of Optical Communications and Networking* 10.1 (Jan. 1, 2018), A27. ISSN: 1943-0620, 1943-0639. DOI: 10.1364/JOCN.10.000A27. URL: https://opg.optica.org/abstract.cfm?URI=jocn-10-1-A27.

[93] Alba P. Vela et al. "BER Degradation Detection and Failure Identification in Elastic Optical Networks." In: *Journal of Lightwave Technology* 35.21 (Nov. 1, 2017), pp. 4595–4604. ISSN: 0733-8724, 1558-2213. DOI: 10.1109/JLT.2017.2747223. URL: http://ieeexplore.ieee.org/document/8022684/ (visited on 04/24/2024).

[94] Génesis Villa et al. "Machine Learning Techniques in Optical Networks: A Systematic Mapping Study." In: *IEEE Access* 11 (2023), pp. 98714–98750. DOI: 10.1109/ACCESS.2023.3312387. URL: https://ieeexplore.ieee.org/document/10242048/.

[95] S. S. Wilks. "Certain Generalizations in the Analysis of Variance." In: *Biometrika* 24.3/4 (1932), pp. 471–494. ISSN: 00063444, 14643510. URL: http://www.jstor.org/stable/2331979 (visited on 08/16/2024).

[96] Huijuan Wu et al. "One-dimensional CNN-based intelligent recognition of vibrations in pipeline monitoring with DAS." In: *Journal of Lightwave Technology* 37.17 (2019), pp. 4359–4366.

[97] Yue Wu et al. "DeepDetect: A cascaded region-based densely connected network for seismic event detection." In: *IEEE Transactions on Geoscience and Remote Sensing* 57.1 (2018), pp. 62–75.

[98] Yue Wu et al. "DeepDetect: A cascaded region-based densely connected network for seismic event detection." In: *IEEE Transactions on Geoscience and Remote Sensing* 57.1 (2018), pp. 62–75.

[99] Xin Xu et al. "Patient-specific method for predicting epileptic seizures based on DRSN-GRU." In: *Biomedical Signal Processing and Control* 81 (Mar. 2023), p. 104449. DOI: 10.1016/j.bspc.2022.104449.

[100] Özal Yıldırım et al. "Arrhythmia detection using deep convolutional neural network with long duration ECG signals." In: *Computers in Biology and Medicine* 102 (2018), pp. 411–420. ISSN: 0010-4825.

[101] Özal Yıldırım et al. "Arrhythmia detection using deep convolutional neural network with long duration ECG signals." In: *Computers in Biology and Medicine* 102 (2018), pp. 411–420. ISSN: 0010-4825.

[102] Shujian Yu, Xiaoyang Wang, and José C. Príncipe. "Request-and-Reverify: Hierarchical Hypothesis Testing for Concept Drift Detection with Expensive Labels." In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, July 2018, pp. 3033–3039. DOI: 10.24963/ijcai.2018/421.

[103] Hongyi Zhang et al. "mixup: Beyond Empirical Risk Minimization." In: *6th International Conference on Learning Representations, ICLR*. 2018.