Proactive Fiber Break Detection based on Quaternion Time Series and Automatic Variable Selection from Relational Data

V. Lemaire[1], F. Boitier[2], J. Pesic[2], A. Bondu[1], S. Ragot[1], F. Clérot[1]

[1] Orange Labs, 2. av. Pierre Marzin, 22307 Lannion Cedex, France
[2] Nokia Bell Labs, Route de Villejust, 91620 Nozay, France

Abstract. We address the problem of event classification for proactive fiber break detection in high-speed optical communication systems. The proposed approach is based on monitoring the State of Polarization (SOP) via digital signal processing in a coherent receiver. We describe in details the design of a classifier providing interpretable decision rules and enabling low-complexity real-time detection embedded in network elements. The proposed method operates on SOP time series, which define trajectories on the 3D sphere; SOP time series are low-pass filtered (to reduce measurement noise), pre-rotated (to provide invariance to the starting point of trajectories) and converted to quaternion domain. After this pre-processing, quaternion sequences are recoded to relational data for automatic variable construction and selection. We show that a naïve Bayes classifier using a limited subset of variables can achieve an event classification accuracy of more than 99% for the tested conditions.

1 Introduction

The requirements for future 5G networks bring new challenges to ensure the reliability of communication infrastructure at affordable cost. While the overall network management is becoming more dynamic and elastic in the metropolitan area thanks to the use of Software Defined Network (SDN) ([32]), the hardware resource allocation to ensure its resilience is becoming more complex and costly. This is especially true at the optical physical layer where the usual resilience mechanism – called dedicated 1+1 optical protection leads to 50% idle network capacity and duplicates a large part of deployed resources to enable path separation ([36]). A solution to avoid this constraint while maintaining a high level of availability is to rely on a proactive restoration mechanism where monitoring triggers, before the failure occurs, the configuration of a new route ([38]).

During the last forty years, several technologies have been developed to use the optical fiber as a sensor, gathered under the Distributed Acoustic Sensing (DAS) field of research ([35]). While most of these technologies have a great sensitivity and demonstrated their potential in pre-warning of infrastructure damage ([1]), the limited maximum distance reach, the compatibility with an optical network in operation and cost still prevent large scale deployment. At

the optical network level, the most probable failure root causes, especially in the metropolitan area, are fiber cuts due to digging activities during civil engineering works ([39],[24]). In this context, the sensors that trigger a proactive restoration do not require the sensitivity reached by DAS systems but their outputs have to accurately discriminate the cause. It was shown in ([37]) that the measurement of the State of Polarization (SOP) of an unmodulated laser light traveling through the optical fiber with a commercial polarimeter could provide an adequate sensor that enables the classification of mechanical events. This was recently confirmed in ([44]), where a low-cost implementation was proposed with only two simple photodiodes and a polarizer with a shorter reach – limited to a single span of fiber between optical amplifiers.

In this paper, we present the results of a joint study conducted by two companies in the framework of a Celtic+ project ([40]). The objective was to design a proactive fiber break detection system based on machine learning with a very strong integration into deployed equipment and the following capabilities: (i) detecting and classifying different kinds of mechanical events; (ii) assuming low cost implementation in networks elements without any additional hardware; (iii) providing interpretable decision results; (iv) allowing real-time hitless restoration mechanisms. In this paper we demonstrate a proactive fiber break detection system based on machine learning implementable as an extension of a high-speed optical receiver. The proposed method operates on SOP time series, which define spherical trajectories; SOP time series are low-pass filtered (to reduce measurement noise), pre-rotated (to provide invariance to the starting point of trajectories) and converted to quaternion domain. After this pre-processing, quaternion sequences are recoded to relational data for automatic variable construction and selection.

The present paper brings several new contributions on top of ([5, 6]) where this work has been partly published. First, a quaternion representation is proposed to exploit the spherical nature of SOP data, avoid overtraining and improve interpretability of decision rules. Second, recoding to relational data with naïve Bayes classification is used to get low-complexity decision rules adapted to real-time implementation as in ([5, 6]); we provide here a comprehensive description and evaluation of a 'data science' solution implemented in a real-time prototype. Third, the SOP database and quaternion pre-processing are made publicly available and a free access to the machine learning software used in this work is provided for the sake of reproducibility (see details at the end of this paper). From a research point of view, we note that the present paper has some limitations (e.g. lack of benchmarking with a reference classification method) which will be addressed in future work.

The paper is organized as follows. Section 2 gives a concise review on related work for time serie classification to explain that we are interested in method which provide simple rules and which can be embedded in network elements. Section 3 describes the fiber break proactive detection principle and the nature of the data used in this work. The real fiber cuts or fiber damages are not rare events but to our knowledge there is no existing process for the on-field

data collection with the required accuracy. Therefore, Section 4 describes (i) how we collect "emulated" data in order to learn[3] on it and (ii) how these data have been pre-processed. Then since our data are represented as time series Section 5 describes the method we used as "classification approach" to discover concise rules if any. Finally, Section 6 presents experimental results and Section 7 describes a proof of concept on a network scenario, before a brief discussion of current limitations and perspectives.

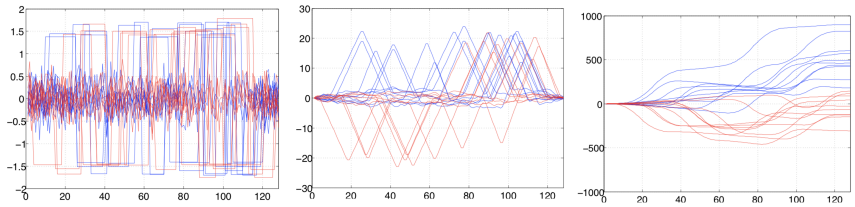
2 Brief Review of Time Series Classification

The field of time series classification represents an extensive literature, and progress has mainly focused on improving the accuracy of classifiers. One interesting point consists in selecting an appropriate representation ([20]) of time series and extracting a table of descriptors in order to train a classifier. Time series classification approaches can be categorized into two classes: *distance-based* and *feature-based* ones. *Distance-based* approaches make a point-wise comparison of entire time series by using similarity measures. As mentioned in ([42]), these techniques are successful when applied on smooth and short time series, but fail for noisy or long time series. In contrast, *feature-based* approaches use features generated from sub-structures of time series in order to make their predictions. For instance, a shapelet is defined as a time series sub-sequences that is maximally representative of a particular class value. The Shapelet Transform approach ([25]) uses the distance to the shapelets as input features for an ensemble of classifier. Generally speaking, ensemble methods are very accurate for time series classification. The key idea of these approaches is to combine different core classifiers and aggregate their predictions using techniques like bagging or majority voting. Diversity in all combined classifiers must be ensured to properly design an ensemble approach. To do this, classifiers of a very diverse nature can be combined. Some approaches ensure diversity by handling multiple representations (derivative, integral ...) of time series (e.g. Elastic Ensemble ([34]), COTE ensemble ([2])).

Numerous data transformation methods for time series have been suggested in the literature such as polynomial, symbolic, spectral or wavelet transformations ([46, 41]). The underlying idea of using data transformation is to look for a new representation space ([3]) where class-characteristic patterns could be easily detected, compared to the original time domain.

The TwoPatterns dataset ([15]) is an interesting illustrative example (see Fig. 1). The objective is to learn a binary classifier from a set of labeled time series. In the original time domain, the class values seem difficult to distinguish (see Fig. 1a). However, the learning task appears easier in the alternative representation obtained by using a cumulative integral transformation (see Fig. 1b). From this representation we count the number of values below -10 for $t \in [20, 100]$ the classification could be solved (the resulting rule in this case could be formed

[3] Note: The machine learning problem addressed in this paper will be defined as a classification problem, for this reason we will mainly use the terminology "classifier" throughout the paper.



(a) (b) (c)

Fig. 1. TwoPattern dataset in time, cumulative and double cumulative integral representations.

as R_1='Count(Signals(t) $\leq$ -10) where t in [20,100]'). At last, using a cumulative double integral transformation makes the classification problem trivial: all curves with value greater than 100 at the last timestamp belong to the blue class, while all other curves belong to the red class (see Fig. 1c). The rule R_1 described above is exactly the one which is automatically discovered by the method describes in Section 5 for the TwoPatterns dataset when using the cumulative integral representation.

This simple example shows that transforming the original time series into an alternative representation space can drastically improve the learned classifier. Moreover this example shows that the use of a adapted representation could help to find simple rules to solve the classification problem. In this paper we will explain in Section 4 how and why the data have been processed to have a adapted representation and Section 5 presents a method coming from the multitable data mining community that discovers simple rules to solve the classification problem.

3 Proactive Fiber Break Detection based on SOP Data

Modern high speed optical communication systems are based on coherent technologies where the light is frequency and polarization multiplexed, and modulated using advanced modulation format where the information is carried not only by the light amplitude but also by its phase ([26]). To recover such a signal from physical impairments occurring during propagation in the fiber and retrieve the sent data, many Digital Signal Processing (DSP) algorithms have been developed and are now commonly used in commercial products.

In this work, we propose to use a coherent receiver which is used in operation to decode data. An extension algorithm is embedded in this receiver to track the SOP, thus avoiding additional dedicated hardware or dedicated optical channel. To limit the required computing power, the proactive detection is based on two key ingredients: (i) rotation speed of SOP and (ii) event classification, which have sufficiently low complexity to be embedded in the coherent receiver.

Figure 2 describes the principle of the proposed approach to collect and classify events at the receiver. In step #1, we take advantage of the DSP block which is required to demultiplex signal polarizations and compensate – among others impairments – for SOP fluctuations occurring over time. For this purpose, Constant Modulus Algorithm (CMA) ([26]) is a widespread method implemented today.

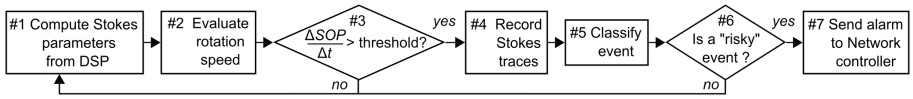


Fig. 2. Flow-chart of the proactive fiber break detection.

The digitized received signal is the projection of the received complex electrical field with an arbitrary polarization on the receiver polarization basis. In order to express the electrical field in the transmitter polarization basis and above all to track this required transformation over time, the CMA algorithm uses four filters (vector) in a butterfly configuration to ensure that the received intensity on each polarization remains constant (a property of the used modulation format where the information is coded in phase only and the intensity is the electrical field modulus). The two input signal components over time are convoluted with the vector of the filter respective to the output basis. The values of the vector are then updated by taking into account the error between the obtained intensity and the expected constant value. This equalizer H can be seen as the inverse of the fiber propagation matrix and allows to compute the SOP expressed in the Stokes coordinates S_1, S_2, S_3 ([23]). For the "y" component, the SOP and the SOP rotation speed can be written:

$$S_0 \propto |H_{xx}|^2 + |H_{yx}|^2 \tag{1a}$$

$$S_1 = \frac{|H_{xx}|^2 - |H_{yx}|^2}{s_0} \tag{1b}$$

$$S_2 = -2\frac{\Re\left(H_{yx}^* H_{xx}\right)}{s_0} \tag{1c}$$

$$S_3 = 2\frac{\Im\left(H_{yx}^* H_{xx}\right)}{s_0} \tag{1d}$$

$$\frac{\Delta \mathrm{SOP}}{\Delta t} = \frac{2}{\Delta t}\sin^{-1}\left(\frac{1}{2}\sqrt{\sum_i (\Delta S_i)^2}\right) \tag{1e}$$

where H_{ij} is the sum of respective filter coefficients from the i-th input polarization to j-th output polarization. In step #2, the SOP rotation speed is tracked using Eq. 1e, where Δt is the time interval between consecutive computed SOP values and ΔS_i the associated Stokes parameter differences with $i = 1, 2, 3$. In step #3, the SOP rotation speed is compared to a threshold-value which can generate a trigger to move to step #4, i.e. the recording of an event during few seconds made of pre-trigger and post-trigger samples. In step #5, this saved event is then sent to the classifier which processes the data and identifies the class of event. Finally, in step #6, if the event is classified as "risky", the receiver raises an alarm through signal messaging to the control plane in step #7. The network controller decides the appropriate action (see Section 7).

4 SOP Data Collection and Pre-Processing

4.1 Data collection

Depending on the operator network size, fiber cuts or damages are likely to happen on a daily to weekly basis. Over the years many studies have been conducted on the occurrence of fiber cut incidents to determine the root causes

for the cut frequency, the impacts on the network quality of service delivery and many other aspects ([24]). Nevertheless, to the best of our knowledge, a database containing SOP variations just before or during the fiber cut does not exist yet. Therefore we provoked in our lab non-destructive mechanical perturbation in order to create a database of emulated events. These emulated events were used for learning and performing the prediction mechanism. In this section we describe the experimental setup we used to emulate the signature of a fiber cut and collect the related SOP events; since the data representation for time series classification is crucial ([20]), we describe method and reasons for coding the SOP time series as quaternions.

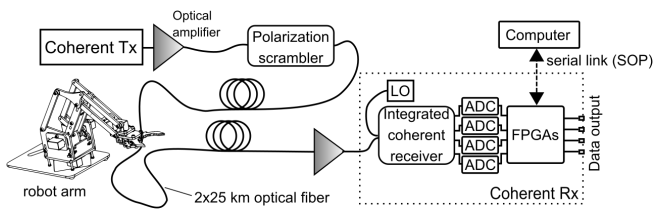


Fig. 3. Experimental setup scheme with programmable mechanical events driven by a robot arm.

Figure 3 shows the experimental setup. A signal carrying pseudo-random data is emitted by coherent transmitter at 28 Gb/s then amplified. After passing through a polarization scrambler which sets a new random polarization state between events, the signal propagates over 2x25 km of optical fiber. The claw of a robot arm controlled by an Arduino, provokes different mechanical events on the fiber in the middle. The fiber output is connected to a last optical amplifier to compensate propagation loss. The receiver board, which is presented in details in ([16]), consists in an integrated coherent receiver followed by four 5-bit Analog-to-Digital Converters (ADCs) operating at a frequency of two samples per symbol followed by FPGAs. All DSP occurs in one FPGA. The CMA aiming at demultiplexing tributaries of each polarization is implemented as the main adaptive 5-tap FIR filter operating on 128 samples in parallel.

The three Stokes parameters are computed accordingly to Eq. 1, rounded to 8-bit signed integers and sent to a computer via a serial interface (see details in ([5])). The computer collects new Stokes parameters at a sampling rate $f_{col} =$ 1920 Hz and the rotation speed threshold in step #3 is set to 0.7 rad/s. The saved event consists of 256 pre-triggered and 7872 post-triggered samples. At the end of the experimental phase we collected 16548 events. Each event consisted of a multivariate time series (MTS) of 3-dimensional data points ($S_1(t)$, $S_2(t)$, $S_3(t)$) coordinates on the polarization sphere (see Fig. 4), with a length of 8128 samples (256 pre-triggered and 7872 post-triggered samples). The robot arm was designed to create four types of events: "bending", "shaking", "small hit" and "up & down" represented respectively by 2873, 1921, 6562, 5195 MTS. Figure 4 shows an example of the evolution of the Stokes parameters as a function of time

during one event and its representation on the Poincaré sphere. From this point the problem of proactive fiber break detection is thus turned into a classification problem of multivariate time series.

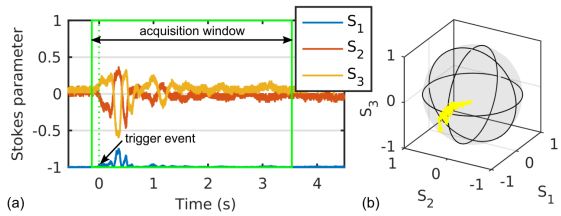


Fig. 4. (a) Example of Stokes parameters $(S1, S2, S3)$ vs time for a "shaking" event. Green rectangle shows the acquisition window. Green dashed line indicates the trigger signal raised in step #3 of Fig. 2. (b) The stokes parameters $(S1, S2, S3)$ represented (yellow points) as a 3D trajectory.

4.2 Pre-processing: low-pass filtering, pre-rotation and conversion to quaternion representation

As a preliminary step, the SOP data is passed through an exponential moving average (with forgetting factor $\alpha = 0.1$) applied independently to each Cartesian coordinate $(S1, S2, S3)$. This is equivalent to low-pass filtering of the input data and smoothing spherical trajectories; this step is commonly used to reduce measurement noise.

In this work we tried to take advantage of the spherical nature of SOP data. Indeed, a SOP time series can be seen as a trajectory on the 3D sphere as shown in Figure 4. In exploratory steps of this work, which are not reported here by lack of space, it was found out that the classification method described in Section 5 resulted in decision rules and variables that could depend on for example the starting point of SOP trajectories[4]. This issue was due to the fact that SOP trajectories collected for learning and testing were produced with a costly and time-consuming process, and the number of trajectories was too limited to prevent the starting point position (or derived statistics such as the average over the MTS) from being informative. These decision rules were not useful, interpretable by experts and could generate over-training which will give bad results on new MTS. Therefore, after several experiments, we considered using quaternions, which are a well-known representation in the fields of computer graphics ([43]) and physics (aerospace, robotics, etc.) ([22]). We used this representation to provide invariance to the starting point of SOP trajectories and to convert XYZ coordinates in a domain which represents incremental rotational movements along spherical trajectories; the conversion to quaternion domain

[4] Invariance to the starting point is quite different from invariance to time scale that could be addressed using dynamic time warping (DTW). Here DTW would not solve the problem of invariance to the starting point.

proved empirically to be efficient in getting variables and decision rules that are more useful and interpretable.

Quaternions (also called hypercomplex numbers or hamiltonians) were introduced in 1843 by Hamilton ([21]) to define a vectorial system generalizing complex numbers. A quaternion q is defined as $q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$, where $(a, b, c, d) \in \mathbb{R}^4$, with the following rules for $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$: $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$. One may also write q as $q = a + \mathbf{q}$, where a is the *scalar* (or real) part of q and $\mathbf{q} = b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ is the *vector* (or imaginary) part of q. Note that $\mathbf{q}$ may be interpreted as a 3D vector by identifying $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$ with three orthogonal Cartesian unit vectors. We do not review basic operations on quaternions the interested reader is referred to ([14, 21, 22]) for more details. We recall that the norm $|\mathbf{q}|$ of a quaternion is: $|q| = \sqrt{a^2 + b^2 + c^2 + d^2}$. When $|q| = 1$, q is said to be a *unit-norm quaternion*.

We used two important properties of quaternions. First, the set of 3D rotations can be mapped to the unit sphere in $\mathbb{R}^4$ under a one-to-two mapping ([14, 43, 22]), namely each 3D rotation matrix maps to two antipodal unit-norm quaternions: q and $-q$, therefore this mapping is not unique. Second, a 3D rotation of angle θ and normalized axis $\mathbf{n}$ (with $|\mathbf{n}| = 1$) is represented by the unit quaternion $q = \cos(\theta/2) + \sin(\theta/2)\mathbf{n}$ or its opposite $-q$. The 3D rotation matrix associated to a unit-norm quaternion $q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ is given by:

$$\mathbf{R}(q) = \begin{pmatrix} 1 - 2(c^2 + d^2) & 2(bc - ad) & 2(ac + bd) \\ 2(ad + bc) & 1 - 2(b^2 + d^2) & 2(cd - ab) \\ 2(bd - ac) & 2(ab + cd) & 1 - 2(b^2 + c^2) \end{pmatrix} \tag{2}$$

Quaternions were applied in this work for two purposes:

- Each SOP trajectory was pre-rotated such that the starting point was brought to the North pole of the 3D sphere. The time series $\mathbf{S}(t) = (S_1(t), S_2(t), S_3(t))^T$, with $t = 0, \ldots, L-1$, defines a trajectory of points lying on the unit polarization sphere. A rotation matrix $\mathbf{R}(q)$ (from Eq. 2) allowing to bring the starting point $\mathbf{S}(0)$ of this trajectory to the North pole $\mathbf{p} = (0, 0, 1)^T$ was computed based on $q = cos(\theta/2) + sin(\theta/2)\mathbf{n}$, where the rotation angle θ is the angle between $\mathbf{S}(0)$ and $\mathbf{p}$, and the rotation axis $\mathbf{n}$ is the (normalized) cross product of $\mathbf{S}(0)$ and $\mathbf{p}$. This pre-rotation by $\mathbf{R}(q)$ was applied to the whole SOP time series $\mathbf{S}(t)$ to obtain $\mathbf{S}'(t) = \mathbf{R}(q)\mathbf{S}(t)$.
- Each (pre-rotated) SOP trajectory was converted to a quaternion time series representing successive rotations from one point to the next one. A straightforward approach would compute for each time instant a quaternion $q(t)$, $t = 1, \cdots, L-1$, representing the incremental rotation from $\mathbf{S}'(t-1)$ to $\mathbf{S}'(t)$, i.e. $q(t) = cos(\Omega(t)/2) + sin(\Omega(t)/2)\mathbf{r}(t)$, where $\Omega(t)$ is the angle between $\mathbf{S}'(t-1)$ and $\mathbf{S}'(t)$ and $\mathbf{r}(t)$ is the (normalized) rotation axis given by their cross product. Due to the non-unique representation of rotations by antipodal unit-norm quaternions, we selected, among the two possible choices $\pm q(t)$, the *canonical* form of unit-norm quaternions having by convention a positive scalar part: $q_C(t) = q(t)$ if $cos(\Omega(t)/2) \geq 0$ and $-q(t)$ otherwise. Moreover, to ensure that the quaternion trajectory followed the shortest

path on the 4D sphere ([43]), the relative angle between successive unit-norm quaternions was used to further select antipodal quaternions as follows: starting with $q_{SP}(1) = q_C(1)$, we set for $t = 2, \cdots, L-1$ $q_{SP}(t) = q_C(t)$ if $q_C(t) \cdot q_{SP}(t-1) \geq 0$ and $-q_C(t)$ otherwise, where $q_1 \cdot q_2$ is the dot product of q_1 and q_2 seen as two 4D vectors.

From now the multivariate time series $(S1(t), S2(t), S3(t))$, which define trajectories on the 3D unit sphere, are converted to quaternion time series $q_{SP}(t) = a(t) + b(t)\mathbf{i} + c(t)\mathbf{j} + d(t)\mathbf{k}$, which can be seen as 4-dimensional vectors $(a(t), b(t), c(t), d(t))$ defining trajectories on the 4D unit sphere, representing incremental rotations from the pre-rotated smoothed SOP data.

5 Time Series Classification Approach

Having an adapted representation of the input multivariate time series (MTS), we may use a method from the literature to build a classifier among the extensive literature (see Section 2 and ([4])). Remind that our requirements are (i) low cost implementation in networks elements without any additional hardware; (ii) providing interpretable decision results if possible using a limited number of concise rules. The scalability is also an issue. Here we decided[5] to turn our time series problem into a relational problem, then apply an existing approach (able to discover rules) to deal with relational data and classification problems. Thus we essentially bring ideas from relational data mining to time series classification.

The time series classification used in this work follows three successive steps: (i) multivariate time series are recoded to relational data (i.e. several linked tables); (ii) a propositionalization approach ([13]) is then used to extract informative aggregate variables from relational data; (iii) a classifier is trained based on the extracted aggregate variables. Once the structure of the relational data is specified, the number of extracted aggregate variables is the only parameter (P below in the paper). This section presents in details this time series classification approach.

This kind of approach, described in ([12, 10, 20]), is fully automatic, scalable and highly robust, with test performance mainly equivalent to train performance. Furthermore, this approach obtained good ranking in classifying trajectories in the FedCSIS challenges in 2015, 2016 and the second place of the AALTD Challenge in 2016 ([45]).

5.1 Encoding multivariate time series as relational data

The first step of the used classification approach consists in gathering the previously computed quaternion sequences $q_{SP}(t) = (a(t), b(t), c(t), d(t))$ within a relational dataset. The "schema" of a relational dataset defines the structure of the included tables, their types (i.e. root or secondary tables) and their links. A large variety of possible schemes exist, and the choice of a particular schema may

[5] Up to now this is an adhoc decision discussed in the last section of this paper

have a significant impact on quality of the learned classifier. As shown in Fig. 5 we consider two possible relational schemas: (i) the *"one-each"* schema includes a root table and four secondary tables (see Fig. 5a) and (ii) the *"all-in-one"* schema that contains a root table and a single secondary table (see Fig. 5b).

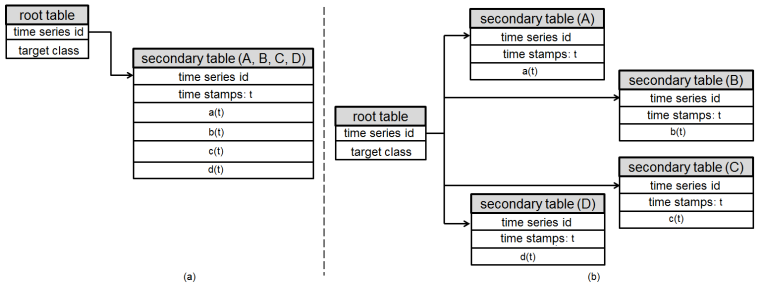


Fig. 5. Relational schemas for encoding multivariate time series as relational data. (a) *all-in-one* schema and (b) *one-each* schema.

These two kinds of schemas correspond to different ways of generating aggregate variables: (i) the *"all-in-one"* schema promotes the generation of "cross-quaternion" variables (i.e. jointly based on the four dimensions of our quaternion representation) (ii) the *"one-each"* schema prioritizes the mining of complex aggregates independently on each dimension of the quaternion representation. During the study we tested the two possibilities but in this paper we only present the result obtained with the *"all-in-one"* schema which gives the best results.

5.2 Automatic variable construction from relational data

The propositionalization approaches consist in learning a model from relational data by flattening the original relational data, which are stored in several linked tables (similarly to databases) ([29]). These approaches come from the field of Relational Data Mining ([17]) and are not usually applied to time series. More precisely, relational data contains at least one root table where each row represents a statistical individual (e.g. 'id' of the time series) and another secondary table containing detail records (eg. the data points of the time series are represented by each row of the secondary table). The specificity of relational data is to involve one-to-many relationships between the tables (eg. a time series contains many data points). The propositionalization problem consists in transforming the relational data into a single attribute-value dataset in order to use regular machine learning methods.

Two kinds of propositionalization approaches can be distinguished (i) *logic-based* methods, such as RSD ([31]), SINUS ([18]), tackle the propositionalization task by constructing first-order logic attributes; (ii) the *database-inspired* methods such as RELAGGS ([28]) apply aggregation functions, such as Min, Max, and Mean in order to generate attributes. The interested reader can find a complete state-of-the-art and a comparative evaluation in ([27]).

In our study, we used a Minimal Description Length (MDL) based propositionalization approach recently presented in ([13]). This approach exploits a Bayesian formalism to generate informative aggregate variables in a supervised way. To the best of our knowledge, this approach is the only one that avoids overfitting problems by regularizing the complexity of the generated variables.

Our relational data (see Fig. 5a) consists of the root table which contains 16548 instances, characterized by two variables: the time series identifiers and the target variable (i.e. class values). The secondary table contains 134 502 144 detailed records (16548 x 8128), described by six variables: the time series identifiers used as a join key, the timestamps of the records and the 4 dimensions of the quaternions values. The number of aggregate variables to be constructed (P) is the only one user parameter. Our relational data is transformed into a regular attribute-value dataset by applying the MDL based propositionalization approach.

5.3 Variable selection and learned classifier

The used MDL approach is able to select the most informative aggregate variables in two ways: i) by filtering uninformative aggregate variables; ii) by finding the most informative and independent subset of variables.

Filtering uninformative variables: The filtering of the aggregate variables is based on previously developed supervised discretization ([8]) and grouping ([7]) methods. This Bayesian approach turns the learning task into a model selection problem. A prior distribution is defined on the model space that exploits the hierarchy of their parameters. In practice, this approach reaches a good trade-off between robust and accurate models. The prior favors simple models with few intervals, and the likelihood favors models that fit the data regardless of their complexity. The aggregate variables are evaluated, one by one, using a specifically designed MDL optimization criterion ([13]). The complexity of the aggregate variables is taken into account by adding a construction cost in the prior. This criterion can be interpreted as a coding length according to information theory. Compression Gain (CG) compares the coding length of the learned model with the empty model that includes a single interval. CG measures the ability of the learned models to compress the training data, despite the additional construction cost. Only the Q variables with a positive CG are retained ($Q \leq P$).

Finding the most informative subset of variables: All the Q informative variables coming from the previous step (after discretization or grouping values) are gathered together and used to learn a Selective Naive Bayes classifier (a Naive Bayes which uses a subset of the Q variables defined by a selective process). The Selective Naive Bayes (SNB) aims to select the most informative and independent subset of variables by using a specifically designed MDL optimization criterion ([9]). The used optimization algorithm consists in a sequence of feed-forward and feed-backward selection steps. These two selection steps are

repeated as long as the optimization criterion reaches a better value. This heuristic is embedded into a multi-start (MS) algorithm, by repeating the feed-forward and feed-backward steps from several random orderings of the variables. At the end, we keep the most probable subset of variables compliant with the naive Bayes assumption, i.e. both informative and independent. This subset contains R aggregate variables, with $R \leq Q \leq P$.

Learning classifier: Finally, the used classifier is a naive Bayes which takes the R selected aggregate variables as an input. As shown in Eq. 3, the naive Bayes classifier ([30]) estimates the distribution of a particular class value C_z conditionally to the input variables x_k.

$$P(C_z|x_k) = \frac{P(C_z)\prod_{j=1}^{J} P(V_j = x_{jk}|C_z)}{\sum_{t=1}^{C}\left[P(C_t)\prod_{j=1}^{R} P(V_j = x_{jk}|C_t)\right]} \tag{3}$$

C is the number of class values to be predicted (in this paper $C = 4$, see Section 4.1). This simple and efficient classifier makes the assumption that the distributions $P(V_j = x_{jk}|C_z)$ are independent. In practice, these conditional distributions are estimated in a frequentist way, by using the previously learned univariate discretization ([8]) and grouping ([7]) models. The denominator of Eq. 3 normalizes the estimated probability by making a sum of the numerator term over all the class values. At the end, the predicted class value given a particular x_k is the one that maximizes the conditional probabilities $P(C_z|x_k)$.

6 Results and Discussion

We trained the naïve Bayes classifier, with the aim to find a function f such as $Y = f(X)$ where Y represents the class, and X is vector of explanatory variables which contains a representation of the time-series $(S_1(t), S_2(t), S_3(t))$ as quaternion time values $q_{SP}(t) = (a(t), b(t), c(t), d(t))$. After the training phase, we can classify new events, i.e. 'unknown events', in order to predict their class. In the experiments, we divided the 16548 events into training and test parts according to the 10-fold cross validation process whatever their class (stratified random sampling).

The vector X is obtained following the steps method described in Section 5 above where: (i) P explanatory variables are constructed (ii) then Q variables (a subset of the P variables) are judged as informative, i.e. they bring information to predict Y, (iii) finally R explanatory variables (a subset of the Q variables) are kept using a forward backward selection mechanism.

The classifier uses a naïve Bayes classifier ([30]) relying on the R variables. Table 1 gives the values of P, Q and R and the classifier performance using two criteria: the Area Under the receiver operating characteristic Curve (AUC) ([19]) and the Accuracy (ACC, the rate of good classification) for the test.

The performance of the classifier increases with P until a stable level where constructing more variables does not bring real improvement. For $P = 1000$

P	Q	R	Train AUC (x100)	Test AUC (x100)	Train ACC	Test ACC
10	7 ± 1.2	6 ± 1	92.36 ± 0.06	88.37 ± 6.61	73.35 ± 0.17	72.65 ± 0.92
100	82 ± 1.8	6.5 ± 0.67	99.96 ± 0.01	99.95 ± 0.03	99.05 ± 0.17	98.96 ± 0.27
1000	882 ± 4.2	4.2 ± 0.40	99.98 ± 0.01	99.96 ± 0.03	99.51 ± 0.07	99.50 ± 0.22
10000	9132 ± 4.7	4.7 ± 0.45	99.99 ± 0.01	99.97 ± 0.04	99.66 ± 0.10	99.62 ± 0.17

Table 1. Number of constructed (P), informative (Q), used variables (R) and classification results obtained (mean ± standard deviation) over the 10 test-folds.

($R = 4.2 \pm 0.4$) the classifier exhibits excellent results (AUC and ACC close to 1 which is for both cases close to the upper bound) and excellent robustness (ratio train/test results close to 1). This last point ensures to have similar performances with unknown events.

Another interesting result is the low number of variables really needed in the classifier (R). This is key to enable a real-time implementation in a commercial transceiver, for which the R variables will be computed online. Moreover, most of them are easy to preprocess online to enable incremental learning ([33]) for fast predictions. We present here four typical examples of found variables (aggregates) of interest:

- 'Sum(b(t)) where $t > 1.9835$'
- 'Sum(d(t)) where $t \in]0.9252, 1.9835]$'
- 'Min(d(t)) where $t > 1.9835$'
- 'Mean(c(t)) where $t \in]0.9252, 1.9835]$'

These variables are easy to interpret. For example the second one based on d indicates the quantity of movement on the d component of the quaternion (the movement on initial $S_3(t)$) for a given period of time of the time series. From this point, we have a classifier with a very good accuracy and a low complexity that can be embedded in coherent receiver for a negligible cost.

These good results and the small number of variables used for classification are due to the proposed global processing chain: (i) a data transformation based on quaternions visibly well suited to the problem, (ii) a representation in multitable form coupled with an efficient search for informative aggregates, and (iii) the use of an interpretable classifier. For future works on SOP signals, the quaternion representation seems to be a very good recommendation

7 Proof of Concept

A proactive mechanism is expected to detect not only fiber breaks but also cases where the cut does not occur – e.g. an excavator near the fiber. In ([38]), it was suggested that a new data route should be established on an upper network layer (the IP layer) to avoid outages due to optical hardware reconfiguration. In ([6]) a new elastic transceiver architecture embedding a classifier similar to the one presented in the previous section was proposed, to enable a seamless optical restoration directly at the optical layer. A key enabler is the ability of this prototype transceiver to emit on two distinct channels and to switch very

quickly the decoding channel. This networking proof of concept was demonstrated in ([6]) and within the concerned companies[6]. This prototype system has been upgraded with the classifier presented in the present paper, to take advantage of the improved performance.

8 Conclusion and Perspectives

From an 'applied data science' point of view, we proposed in this work to use a real-time coherent receiver coupled with machine learning to monitor mechanical events on an optical fiber, to proactively detect fiber breaks. We experimentally validated that multivariate time series (trajectories of SOP of light propagating through an optical fiber) could be represented efficiently using quaternions. Our system exhibits an accurate event classification with more than 99%. We developed a proof of concept which shows that it is possible to detect and classify different kinds of events in real time to have real-time seamless optical restoration mechanisms. Furthermore the small number of rules found by the classifier are concise, clearly interpretable. Nokia Bell Labs is currently studying the implementation of the method in the field.

From a research point of view some work remains to be done. We need to (i) see if building a multivariate time series classifier directly (without the multitable data mining approach) could give concise and interpretable rules, (ii) do a thorough comparison against a wide range of time series representations (e.g. variants of quaternion representations or other ways to represent spherical data).

Note on reproducibility: The following files can be downloaded at `https://bit.ly/2WXu19V`: (i) the SOP dataset (multitable representation, 700MB zipped in split files, 10GB unzipped). This dataset is released for the sake of verification and investigations of alternative methods; (ii) the Python module to convert the initial data to quaternions. A provisional license of the Khiops software implementing all the elements of Sections 5.2 and 5.3 can be obtained for free from ([11]).

References

1. Adewuyi, A.P., Wu, Z., Serker, N.K.: Assessment of vibration-based damage identification methods using displacement and distributed strain measurements. Struct. Health Monit. **8**(6), 443–461 (2009)
2. Bagnall, A., Lines, J., Hills, J., Bostrom, A.: Time-series classification with COTE: the collective of transformation-based ensembles. IEEE Trans. Knowl. Data Eng. **27**(9), 2522–2535 (Sept 2015)
3. Bagnall, A., Davis, L., Hills, J., Lines, J.: Transformation based ensembles for time series classification. In: Proc. 12th SDM (Apr 2012)
4. Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.: The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Min. Knowl. Disc. **31**(3), 606–660 (2017)

[6] Not named here for blind review.

5. Boitier, F., Lemaire, V., Pesic, J., Chavarría, L., Layec, P., Bigo, S., Dutisseuil, E.: Proactive fiber damage detection in real-time coherent receiver. In: Proc. ECOC (2017)
6. Boitier, F., Pesic, J., Lemaire, V., Dutisseuil, E., Tolosa, J.M.E., Jennevé, P., Moing, N.L., Mardoyan, H., Layec, P.: Seamless optical path restoration with just-in-time resource allocation leveraging machine learning. In: Proc. ECOC. p. Demo Session (2018)
7. Boullé, M.: A grouping method for categorical attributes having very large number of values. In: Perner, P., Imiya, A. (eds.) Proceedings of the Fourth International Conference on Machine Learning and Data Mining in Pattern Recognition. vol. 3587, pp. 228–242 (2005)
8. Boullé, M.: MODL: a Bayes optimal discretization method for continuous attributes. Mach. Learn. **65**(1), 131–165 (2006)
9. Boullé, M.: Compression-based averaging of selective naive Bayes classifiers. J. Mach. Learn. Res. **8**, 1659–1685 (2007)
10. Boullé, M.: Tagging fireworkers activities from body sensors under distribution drift. In: Proc. Federated Conf. Comput. Sci. Inf.Syst. pp. 389–396 (2015)
11. Boullé, M.: Khiops: outil d'apprentissage supervisé automatique pour la fouille de grandes bases de données multi-tables. In: Extraction et Gestion des Connaissances. pp. 505–510 (2016), `http://www.khiops.com`, [available at `http://www.khiops.com`]
12. Boullé, M.: Predicting dangerous seismic events in coal mines under distribution drift. In: Ganzha, M., Maciaszek, L., Paprzycki, M. (eds.) Proc. Federated Conf. Comput. Sci. Inf.Syst. pp. 221–224 (2016)
13. Boullé, M., Charnay, C., Lachiche, N.: A scalable robust and automatic propositionalization approach for bayesian classification of large mixed numerical and categorical data. Mach. Learn. (2018)
14. Casteljau, P.D.: *Les quaternions*. Dunod (1987)
15. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The UCR time series classification archive (July 2015), `www.cs.ucr.edu/~eamonn/time_series_data/`
16. Dutisseuil, E., Tanguy, J.., Voicila, A., Laube, R., Bore, F., Takeugming, H., de Dinechin, F., Cerou, F., Charlet, G.: 34 Gb/s PDM-QPSK coherent receiver using SiGe ADCs and a single FPGA for digital signal processing. In: Proc. OFC. p. OM3H.7 (2012)
17. Džeroski, S.: Relational data mining. In: Data Min. Knowl. Disc. Handbook, pp. 887–911. Springer (2009)
18. Dzeroski, S., Lavrac, N.: Inductive logic programming: techniques and applications. Prentice Hall, New York (1994)
19. Fawcett, T.: ROC graphs: Notes and practical considerations for researchers. Tech. Rep. HPL-2003-4, HP Laboratories (2004)
20. Gay, D., Guigourés, R., Boullé, M., Clérot, F.: Feature extraction over multiple representations for time series classification. In: International Workshop NFMCP held at ECML/PKDD. pp. 18–34 (2013)
21. Hamilton, W.R.: On a new species of imaginary quantities connected with a theory of quaternions. Proc. R. Ir. Acad. **2**, 424–434 (1843)
22. Hanson, A.J.: Visualizing Quaternions. Morgan Kaufmann Publishers (2006)
23. Hauske, F.N., Kuschnerov, M., Spinnler, B., Lankl, B.: Optical performance monitoring in digital coherent receivers. J. Lightwave Technol. **27**(16), 3623–3631 (2009)
24. Hayford-Acquah, T., Asante, B.: Causes of fiber cut and the recommendation to solve the problem. IOSR J. Electron. Commun. Eng. **12**, 46–64 (2017)

25. Hills, J., Lines, J., Baranauskas, E., Mapp, J., Bagnall, A.: Classification of time series by shapelet transformation. Data Min. Knowl. Disc. **28**(4), 851–881 (2014)
26. Kikuchi, K.: Fundamentals of coherent optical fiber communications. J. Lightwave Technol. **34**(1), 157–179 (2016)
27. Krogel, M.A., Rawles, S., Železnỳ, F., Flach, P.A., Lavrač, N., Wrobel, S.: Comparative evaluation of approaches to propositionalization. In: Inductive Logic Programming. pp. 197–214. Springer (2003)
28. Krogel, M.A., Wrobel, S.: Transformation-based learning using multirelational aggregation. In: Inductive Logic Programming. pp. 142–155. Springer (2001)
29. Lachiche, N.: Propositionalization, pp. 812–817. Springer US, Boston, MA (2010)
30. Langley, P., Iba, W., Thompson, K.: An analysis of Bayesian classifiers. In: Proceedings of the Tenth National Conference on Artificial Intelligence. pp. 223–228. AAAI'92 (1992)
31. Lavrač, N., Železnỳ, F., Flach, P.A.: RSD: Relational subgroup discovery through first-order feature construction. In: Inductive Logic Programming. pp. 149–165. Springer (2002)
32. Layec, P., Dupas, A., Verchère, D., Sparks, K., Bigo, S.: Will Metro networks be the playground for (true) elastic optical networks? J. Lightwave Technol. **35**(6), 1260–1266 (2017)
33. Lemaire, V., Salperwyck, C., Bondu, A.: A survey on supervised classification on data streams. Lect. Notes Bus. Inf. P. (2015)
34. Lines, J., Bagnall, A.: Time series classification with ensembles of elastic distance measures. Data Min. Knowl. Disc. **29**(3), 565–592 (2015)
35. Liu, X., quan Jin, B., Bai, Q., Wang, Y., Wang, Y.: Distributed fiber-optic sensors for vibration detection. In: Sensors (2016)
36. Nokia white paper: Advances in optical layer restoration (2017), `https://www.nokia.com/blog/optical-layer-restoration-improving-efficiency/`
37. Pesic, J., Le Rouzic, E., Brochier, N., Dupont, L.: Proactive restoration of optical links based on the classification of events. In: Proc. ONDM. pp. 1–6 (2011)
38. Pesic, J., Meuric, J., Le Rouzic, E., Dupont, L., Morvan, M.: Proactive failure detection for WDM carrying IP. In: Proc. IEEE INFOCOM. pp. 2971–2975 (2012)
39. Pesic, J.: Study of the mechanisms associated with the preventive network restoration in fiber optic core networks. Ph.D. thesis, Université de Bretagne-Sud (2012)
40. Project SENDATE-Tandem: Secure networking for a data center cloud in europe - tailored network for data centers in the metro, `https://www.celticnext.eu/project-sendate-tandem/`
41. Rodríguez, J.J., Alonso, C.J., Boström, H.: Learning first order logic time series classifiers: Rules and boosting. In: European Conference on Principles of Data Mining and Knowledge Discovery. pp. 299–308. Springer (2000)
42. Schäfer, P., Leser, U.: Fast and accurate time series classification with WEASEL. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 637–646 (2017)
43. Shoemake, K.: Animating rotation with quaternion curves. ACM SIGGRAPH Computer Graphics **19**(3), 245—-254 (1985)
44. Simsarian, J.E., Winzer, P.J.: Shake Before Break: Per-Span Fiber Sensing with In-Line Polarization Monitoring. In: Proc. OFC. p. M2E.6 (2017)
45. Time-series classification challenge: Workshop advanced analytics and learning on temporal data at ECML (2016), `https://aaltd16.irisa.fr/challenge/`
46. Warren Liao, T.: Clustering of time series data – A survey. Pattern Recognition **38**(11), 1857–1874 (2005)