Scoring Message Stream Anomalies in Railway Communication Systems

Lucas Foulon, Serge Fenet, Christophe Rigotti, Denis Jouvin

Scoring Message Stream Anomalies in Railway Communication Systems

Lucas Foulon[1], Serge Fenet[2], Christophe Rigotti[3], and Denis Jouvin[4]

[1] Univ Lyon, CNRS, LIRIS, UMR5205, F-69621, Villeurbanne, France
et SNCF Mobilit, DSI Production Ferroviaire, F-69393, Lyon, France
lucas.foulon@sncf.fr
[2] Univ Lyon, Université Claude Bernard Lyon 1, CNRS,
LIRIS, UMR5205, F-69621, Villeurbanne, France
serge.fenet@liris.cnrs.fr
[3] Univ Lyon, INSA-Lyon, CNRS, INRIA,
LIRIS, UMR5205, F-69621, Villeurbanne, France
christophe.rigotti@insa-lyon.fr
[4] SNCF Mobilit, DSI Production Ferroviaire, F-69393, Lyon, France
denis.jouvin@sncf.fr

Abstract. This work focuses on the detection of anomalies in a railway communication system. We present the detection method as well as the architecture of the system supporting it. The method is based on a preliminary stage to collect and aggregate the data. It then combines the use of a multidimensional indexation tree (i.e., iSax tree) to store time series, and the computation of an anomaly detection score (i.e., CFOF score). We propose a fast estimation of this anomaly score, based on the information stored in the indexation tree, so that the score can therefore be computed on the fly over the stream of messages in the communication system. We show by mean of experiments that this estimation is close to the exact score. We describe the platform that has been implemented, and we show that it effectively support abnormal behaviour detection in the real stream of messages within the communication system of the French Railway Company (SNCF).

1 Introduction

Detecting abnormal behaviours in numerical data streams is an important task, necessary in a wide range of domains. Such behaviours, called anomalies, novelties, or outliers, tend to deviate from the main dynamics as if they were generated by a different underlying mechanism [1]. In the current context of exponentially growing volume of numerical data coming from a wide panel of applications, anomaly detection represents a challenge of ever-increasing importance.

The French Railway Company (SNCF) produces and processes in real time, in its information system, a large amount of heterogeneous data. It operates the national rail traffic, including the high-speed rail network (TGV), and its functions include railway services for passengers and freight, as well as railway

maintenance and signalling. Some data, like for example real time passenger information including upcoming departures, disturbances, etc., originate from ground local information systems. Others, like geolocation data, remote administration, train state, etc., are generated on board, produced by the so-called communicating trains. All these data are transferred throughout the main information system by messages, generating traces handled mainly using the ELK Stack [2], an open source software platform including the Elasticsearch, Logstash and Kibana tools[5].

In the industrial context of the SNCF company, the aim of this work is to detect abnormal behaviours in the stream of message traces. The main contribution of this paper is twofold. Firstly, it introduces the detection method based on aggregation, indexation and scoring to assess the current state of the system with respect the past states. Secondly, it presents the whole platform based on ELK Stack and supporting the method. The software has been implemented, and tested on real data. These experiments show that the approach is effective and efficient to detect abnormal behaviours in the communication system of SNCF. In addition, when compared to the anomalies detected by the Machine Learning toolbox of ELK Stack, it gives more information about the anomaly durations. This is very important in an industrial context, since this provides evidences to know if the system has returned to a normal state or not.

In order to compare the recent observed parts of the message stream to the past states, we use the CFOF (Concentration Free Outlier Factor) anomaly detection score recently proposed by Angiulli [3]. This score is the only one for which the robustness to the curse of dimensionality has been observed experimentally and formally proven. Unfortunately its existing computation methods are not suitable for data stream, and a technical contribution of our work is to efficiently and closely approximate the CFOF score over this kind of data. This fast estimation is performed by taking advantage of iSax trees [4, 5], that are multidimensional indexing structures developed for time series.

The rest of the paper is organised as follows. The next section presents the related work. The method is presented in Section 3 and the system architecture is described in Section 4. The results are presented and discussed in Section 5, and we conclude with a summary in Section 6.

2 Related Work

New challenges are emerging in the intelligent transport literature, related in particular to the use of data mining and simulation-based solutions. With the fast development of connected transport, many axis of research focus on improving the quality of service and on reducing the maintenance costs in this context. Successful applications of data mining approaches have been recently reported, as for instance, the detection of defective rail anchors investigated by Khan et al. [6], the wavelet-based identification of rail surface defects using images

[5] Elastic, Elasticsearch, Logstash and Kibana are trademarks of Elasticsearch BV, registered in the U.S. and in other countries.

presented by Molodova et al. [7], or the detection of illegal pickups using GPS traces reported by Yin et al. [8]. In this paper, we consider the problem of detecting anomalies in a railway communication system. Anomaly detection is a very active research area with many different applications, as for instance the recent work of Abàmoff et al. [9] to help diabetic retinopathy diagnosis using convolutional neural networks. There are a multiple methods applied to various domains [10, 11]. The main approaches are quickly recalled hereafter.

2.1 Different methods

There are two main families of anomaly detection methods: supervised and unsupervised. One of the representative approaches among the unsupervised methods is, for instance, the isolation forests used by Ting et al. [12], where data that can be isolated easily (by several decision trees) are considered as abnormal. Another more recent approach, proposed in [13], is distributed in the widely use ELK software solution. It is based on Bayesian methods to build a statistical model of the system state, and is available in the Machine Learning toolbox of ELK.

For the supervised methods, typical approaches are the one of Mukkamala et al. [14] that uses a support vector machine classifier in order to detect intrusions, and also patterns/rules-based methods as the work by Li et al. [15] to detect objects having abnormal trajectories. The main difference between supervised and unsupervised approaches is that the supervised ones require a training dataset containing objects that are already labelled as normal or abnormal, and in most applications this dataset needs to be large and representative of many of the possible normal and/or abnormal objects that could be encountered. In our case, the data are not labelled and thus the method must be unsupervised. Let us then focus on the most widely used family of approaches: the proximity based anomaly detection methods.

2.2 Proximity based methods

These methods have a lot of variants, and can be split in three subfamilies [16]: clustering based, density based, and distance based methods. Clustering based methods assess if an object belongs or not to a cluster. The object is considered abnormal if it is not sufficiently close to any of the clusters [14]. Density based methods include techniques that take into account the density distribution of objects in their representation space like the LOF method proposed by Breunig et al. [17]. Distance based methods are the most prevalent, and simply use the distances between an object and objects in its neighborhood to compute an anomaly score [18].

Recently, Angiulli proposed CFOF [3], a new distance based score to detect anomalies. The main advantage of this score is that it can be applied in high dimensional space, i.e., when object are described by many features. Thus, it could be well adapted to handle sequence of measures (where each measure is a dimension). However, the existing methods to compute this score [3] are not

designed to process streaming data as it is the case in our railway communication system.

3 Method

This method aims to detect abnormal patterns in sequences containing the counts of messages at a crucial node in the information system. Anomaly detection in such sequences of measures is a classical task, that is usually performed by comparing news sequences to a base of sequences of reference and by computing an anomaly score [16]. However, in general, this approach is adapted only for small sequences, since, as shown in [3], when the number of dimensions (i.e., the size of the sequences) increases, the anomaly scores suffer from the concentration phenomenon. This phenomenon is part of the so called curse of dimensionality problem, and limits the ability to distinguish between normal and abnormal sequences. In order to overcome this limitation, in this work, we use the CFOF score, proposed by [3], and that has been proven not to concentrate when dimensionality increases. [3] proposed an efficient technique for calculating the CFOF score by sampling, where the scores of all the objects of a sample are computed with respect to all the other objects of the same sample. This technique takes advantage of a the factorisation of the necessary operations within each sample. The quality of the approximation depends on the size of the sample, and this technique is well suited when one wishes to calculate the scores of all the objects of a database. However, it is not adapted when one wants to compute only the score of a new object against a reference history. So, we propose to take advantage of the properties of the iSAX tree structure [4, 5] to efficiently compute a close approximation of the CFOF score of new sequence in a data stream.

In this section, we first describe how we build the sequences from the raw data gathered from the SNCF information system. Then, we briefly recall the definition of the CFOF score, and present the approximation scheme of this score using an iSAX tree.

3.1 From raw streams to iSAX data

The main component of the information system, called CanalTrain, manages the exchange of all digital information between ground-stations and mobile equipment. When the behaviour of any of the routing component involved in this essential service becomes abnormal, serious malfunctions can not only disrupt this central component, but also indirectly impact other services. The raw stream of data is too detailed and voluminous to be directly analysed. As we are interested in the global health of the communication system, we extract a simplified description of the dynamics of the stream from the available raw data. This process is described in figure 1.

The raw data flow is steadily gathered from the various collection points implemented in the system, and made available through a Logstash service (the ELK component that collects and transforms data). The content and semantics

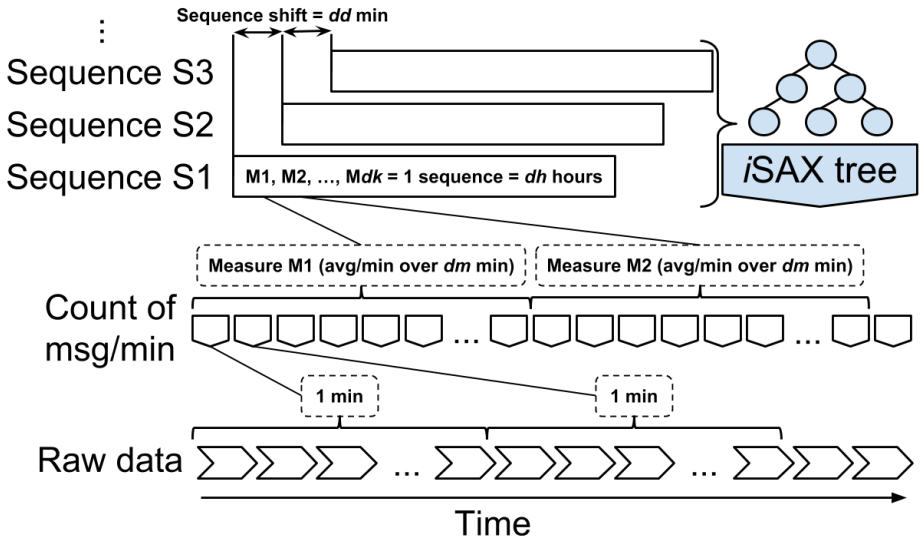

Fig. 1: Extraction process, from raw data to sequences.

of the raw messages are ignored and the message rate is monitored by counting the number M of messages per minute. Every dm minutes, an average value of M is computed. A series of dk consecutive average rate values $M_1, M_2, \ldots, M_{dk}$ is called a *sequence*, than spans a time $dh = dk \times dm$. Such sequences are generated every dd minutes and can overlap over time. A collection of past sequences will form the objects of reference inserted in the iSAX tree, and an approximation of the CFOF anomaly score will be computed for the new sequences constructed from the stream. In the results presented Section 5, the sequences are built with $dm = 15$ minutes, $dk = 24$ (and thus $dh = 6$ hours), and $dd = 30$ minutes.

3.2 Formal definition of the CFOF score

By using an iSAX tree and its space-structuring properties, we are able to accurately approximate the anomaly score of a new series that is not already indexed into the tree. This method will be presented in detail later, but let us first recall more precisely the definition of the CFOF score.

Defined by [3], the CFOF score of an object q is computed by taking into account a set of neighboring objects $\mathcal{R}$. One must first define the minimal neighborhood size k_m such that the object q is among the k_m nearest neighbors of at least a fraction ϱ of all the objects in $\mathcal{R}$.

CFOF(q), the CFOF score, is equal to the value k_m normalized by the size of $\mathcal{R}$. One can see that the value of the score is controled by the parameter $\varrho \in [0; 1]$. Defining the ratio of the objects of $\mathcal{R}$ that must include q in their neighborhood of size k_m, it can therefore control the stringency of the score: a high value of ϱ will increase the value of CFOF score for even small anomalies.

We explain in Section 5 how this parameter can be tuned in order to adapt the score to the intended applications.

Let us now define the score in a more formal way. Let us define $\mathrm{nn}_k(x)$ as the k^{th} nearest neighbor of an object x (meaning that only $k-1$ objects are closer to x than the object $\mathrm{nn}_k(x)$). We can then define $\mathrm{NN}_k(x)$ as the set of the k nearest neighbors of x, i.e. the set of all the objects such that $\{\mathrm{nn}_i(x) \mid 1 \leq i \leq k\}$. If $\mathrm{N}_k(x)$ is the number of objects that contain x among their k nearest neighbors, i.e., $\mathrm{N}_k(x) = |\{y \mid x \in \mathrm{NN}_k(y)\}|$, then the CFOF score of an object q is defined as: $\mathrm{CFOF}(q) = \min\{k/|\mathcal{R}| : \mathrm{N}_k(q) \geq \varrho \times |\mathcal{R}|\}$

Being fundamentally defined by taking into account the local structure of the neighborhood of each object, the CFOF score is, to the best of our knowledge, the only one that has formally been shown to be resistant to the distance concentration phenomenon in high-dimensional data [3].

3.3 Method to approximate the real CFOF score

We propose in this section a method to compute an approximation of the exact CFOF score in order to apply it to our specific industrial context. The objects to score are considered as multidimentional objects described by sequences of values of the same length dk. They are stored in a space-structured iSAX tree [4, 5].

This tree performs traditional multi-dimensional indexing based on the similarity of objects, but also exhibits additional properties that we exploit to compute the anomaly score:

1. No leaf of the tree is overlapping with any other, so that each region of the space is represented by exactly one leaf. This property let us pre-compute statistics about the distribution of the objects over the whole space covered by the tree.
2. Due to the indexing mechanism, we can use the distances between objects as boundaries during searches in the tree: for any possible object p, we can quickly derive, for any node $\mathcal{N}$ of the tree, a lower bound of the distance between p and the closest object to p indexed in the subtree rooted at $\mathcal{N}$. This property is used during the CFOF score computation to prune the search space. We can therefore avoid the detailed exploration of areas containing objects for which we can be sure they are not contained in the neighborhood of interest.

To compute the CFOF score of any new object q in relation with the objects already indexed in the iSAX tree, the crux is to obtain the rank of q in the neighborhood of p for each object p stored in the tree. This rank is 1 if q is the closest neighbor of p, 2 if q if the second closest neighbor, and so on. Denoted $v\text{-}rank_p(q)$, this rank is more precisely defined, as described in Section 3.2, as the value k such that $\mathrm{nn}_k(p) = q$. When the ranks are known for all the objects of reference p, then the computing $\mathrm{CFOF}(q)$ can be done quite directly as follows.

We maintain a list $l_{v\text{-}rank}$ containing the values $v\text{-}rank_p(q)$ sorted in ascending order. Let $\lceil x \rceil$ denotes the rounding up of a number x. If we consider the element in $l_{v\text{-}rank}$ at index $\lceil \varrho \times |\mathcal{R}| \rceil$, then the value of this element is the minimal neighborhood size k_m, such that q belongs to the k_m nearest neighbors of at least a fraction ϱ of the objects in $\mathcal{R}$. We therefore possess the main information to compute the CFOF score CFOF(q) that is the value of k_m normalized by the size of $\mathcal{R}$: CFOF(q) = $l_{v\text{-}rank}[\lceil \varrho \times |\mathcal{R}| \rceil]/|\mathcal{R}|$.

However, in order to compute the rank $v\text{-}rank_p(q)$, we must first obtain the number of objects of reference r such that $distance(p, r) \leq distance(p, q)$. We can do this efficiently by avoiding to count each object r one by one: $v\text{-}rank_p(q)$ can be approximated using the distributions of the distances in the different regions of space that are not empty. We exploit the fact that the Euclidean norm of vectors that have normally distributed components follow a χ distribution. Since this distribution tends toward a normal distribution when the number of dimensions of the space increases, we can therefore use a cumulative normal distribution function of the distance for each leaf $\mathcal{N}_l$, in order to approximate the number of objects r of the leaf for which $distance(p, r) \leq distance(p, q)$.

Let $\Phi_{\mu,\sigma}(x)$ be the cumulative distribution function of the normal distribution $\mathcal{N}(\mu, \sigma^2)$. For a leaf $\mathcal{N}_l$ and an object p, let $\tilde{\mu}$ and $\tilde{\sigma}$ be approximations of the mean and of the standard deviation of the distance between p and the objects in $\mathcal{N}_l$. Then, the fraction of the objects of leaf $\mathcal{N}_l$ that are located within a distance to p that is less than or equal to $distance(p, q)$ can be approximated by $\Phi_{\tilde{\mu},\tilde{\sigma}}(distance(p, q))$. This is the approximation used by the algorithm described in the next section.

3.4 Search process into the *i*SAX tree

In order to implement the efficient search method decribed above, we must first compute and store the numerical approximations of $\tilde{\mu}$ and $\tilde{\sigma}$ for each object of reference with respect to each leaf. The algorithm itself is independant of the method used to perform these approximations, and several methods can be envisionned. We present in Section 3.5 the method we used.

Let us denote $\widetilde{\texttt{dist}}(\mathcal{N}_l, p)$ and $\tilde{\sigma}_{\mathcal{N}_l}(p)$ the value of $\tilde{\mu}$ and $\tilde{\sigma}$ that will be computed for an object of reference p and a leaf $\mathcal{N}_l$. The method also requires the root $\mathcal{N}_{root}$ of the iSAX tree, the set $\mathcal{R}$ of objects of reference (that can be retrieve directly from the leaves of the tree) and the CFOF sensitivity parameter ϱ. The approximation method is described in detail by Algorithm 1.

Given on object of reference p, the algorithm first build an approximation of $v\text{-}rank_p(q)$ by performing a traversal of the tree to count the number of objects that are closer to p than q, that is then inserted in a ascending order-sorted list (line 19). This list is used to get an approximation of $CFOF(q)$ (line 20), as detailed Section 3.3.

The traversal of the tree for a given p is performed by the loop starting line 6. The nodes remaining to visit are stored in the list $list_{\mathcal{N}}$, from which the current node is removed once it has been visited (line 7). The algorithm uses the bounds `minDist` and `maxDist` (derived from the indexing structure of the iSAX tree)

on the distances between p and the objects contained in the subtree rooted at the current node.

Two cases can thus stop the exploration of the subtree:

1. If the minimal distance between p and the objects contained in the current subtree is greater than the distance between p and q (line 8), then no object that can count for the computing of $v\text{-}rank_p(q)$ can exist in this subtree.
2. If the maximal distance is less than the distance between p and q (line 10), then all the objects in the current subtree are closer to p than q is. We can therefore directly add the number of object contained in this current subtree (denoted $nbObj(\mathcal{N}_{current})$ and stored in each node) to *v-rank*.

Two cases must be taken into account while searching into a subtree:

1. If the current node is an internal node (line 16), we keep track that its children must be explored by adding them to the list of nodes to visit $list_{\mathcal{N}}$.
2. If the current node is a leaf (line 12), then the number of objects in the leaf that are closer to p than q is approximated with the cumulative normal distribution $\phi_{\tilde{\mu},\tilde{\sigma}}$.

Algorithm 1: Computation of an approximation of $CFOF(q)$ in an *i*SAX tree.

Data: Object q, set $\mathcal{R}$ of objects of reference, root $\mathcal{N}_{root}$ of the *i*SAX tree, values $\widetilde{\texttt{dist}}(\mathcal{N}_l, p)$ and $\tilde{\sigma}_{\mathcal{N}_l}(p)$, CFOF parameter ϱ

```
1  l_{v-rank} ← ∅
2  forall p in R do
3  |  v-rank ← 0
4  |  dist ← distance(p, q)
5  |  list_N ← [N_root] ;                          // Remaining nodes to visit
6  |  while list_N ≠ ∅ do
7  |  |  N_current ← list_N.pop()
8  |  |  if minDist(p, N_current) > dist then
9  |  |  |  Do nothing, no need to explore the subtree rooted at N_current
10 |  |  else if maxDist(p, N_current) ≤ dist then
11 |  |  |  v-rank ← v-rank + nbObj(N_current)
12 |  |  else if N_current is a leaf then
13 |  |  |  μ̃ ← dist~(N_current, p)
14 |  |  |  σ̃ ← σ̃_{N_current}(p)
15 |  |  |  v-rank ← v-rank + Φμ̃,σ̃(dist) * nbObj(N_current)
16 |  |  else
17 |  |  |  forall N in N_current.children do
18 |  |  |  |  Add N to list_N
19 |  Insert v-rank in l_{v-rank} in ascending order
20 return l_{v-rank}[⌈ϱ × |R|⌉]/|R|
```

3.5 Estimation process of $\widetilde{\texttt{dist}}(\mathcal{N}, p)$ and $\tilde{\sigma}_{\mathcal{N}}(p)$

We briefly explain here how we compute the estimations of $\tilde{\mu}$ and $\tilde{\sigma}$: $\widetilde{\texttt{dist}}(\mathcal{N}, p)$ and $\tilde{\sigma}_{\mathcal{N}}(p)$ respectively. They can be computed in different ways, but must be accurate enough to ensure an accurate approximations of the CFOF score. All the results presented in Section 5 were produced with this method.

Estimation of $\widetilde{\texttt{dist}}(\mathcal{N}, p)$ For any object p and any given node $\mathcal{N}$ containing a set of objects, then $\widetilde{\texttt{dist}}(\mathcal{N}, p)$ is the quadratic mean:

$$\widetilde{\texttt{dist}}(\mathcal{N}, p) = \sqrt{\frac{1}{|\mathcal{N}|} \times (\sum_{r \in \mathcal{N}} |r - p|^2)}$$

If C is considered as the center of mass of the objects in $\mathcal{N}$ (all having a unitary mass), then the *Huygens* theorem states that:

$$\sum_{r \in \mathcal{N}} |r - p|^2 = \sum_{r \in \mathcal{N}} |r - C|^2 + |\mathcal{N}| \times |C - p|^2$$

and so:

$$\widetilde{\texttt{dist}}(\mathcal{N}, p) = \sqrt{\frac{1}{|\mathcal{N}|} \times (\sum_{r \in \mathcal{N}} |r - C|^2 + |\mathcal{N}| \times |C - p|^2)} \tag{1}$$

In this formula, the sum $\sum_{r \in \mathcal{N}} |r - C|^2$ can be precomputed and stored in each leaf. So, when Algorithm 1 requires the value of $\widetilde{\texttt{dist}}(\mathcal{N}, p)$, we only need to compute the distance $|C - p|^2$ in order to get $\widetilde{\texttt{dist}}(\mathcal{N}, p)$.

Estimation of $\tilde{\sigma}_{\mathcal{N}}(p)$ We compute a weighted sum of all the standard deviations along each dimension, that assign more importance to the dimension where p is far from C. Let $\mathcal{D}$ be the number of dimension of the space in which each object is embedded, let $\sigma_1, \sigma_2, \ldots, \sigma_{\mathcal{D}}$ be the standard deviations of the coordinates of the objects stored in $\mathcal{N}$ for each dimension. Let $(p_1, p_2, \ldots, p_{\mathcal{D}})$ (resp. $(C_1, C_2, \ldots, C_{\mathcal{D}})$) be the coordinates of p (resp. of C). Then the value $\tilde{\sigma}_{\mathcal{N}}(p)$ is:

$$\tilde{\sigma}_{\mathcal{N}}(p) = \sum_{d=1}^{\mathcal{D}} (\sigma_d * \frac{|C_d - p_d|}{\sum_{i=1}^{\mathcal{D}} |C_i - p_i|})$$

4 System Architecture

Nearly 5 billion messages are exchanged every day within the SNCF information system. All are handled by the central plateform CanalTrain, and are redistributed between trains and other software components providing additional

services, like for instance the Geomobiles plateform to geolocalize trains. The ELK software is used to monitor these messages and to ensure that they transit properly within the information system. This global architecture is illustrated in figure 2. ELK is composed of three tools: Elaticsearch, Logstash and Kibana. Elasticsearch is used to index and store data in the JavaScript Object Notation (JSON) format. Logstash is a tool used to collect and forward data and events (also named traces) to Elasticsearch – in our case, Logstash creates a trace for each new message arriving in CanalTrain. Kibana, the third tool of ELK suite, is a visualization tool used to create dashboards. Elasticsearch is used to build and store a monthly index of all the traces created by Logstash. SNCF supervisors create dashboards to visualise statistics and features (histograms, scatter plots, ...) related to these traces. The central indicator is the number of messages received per minute, the measure on which we perform the anomaly detection.

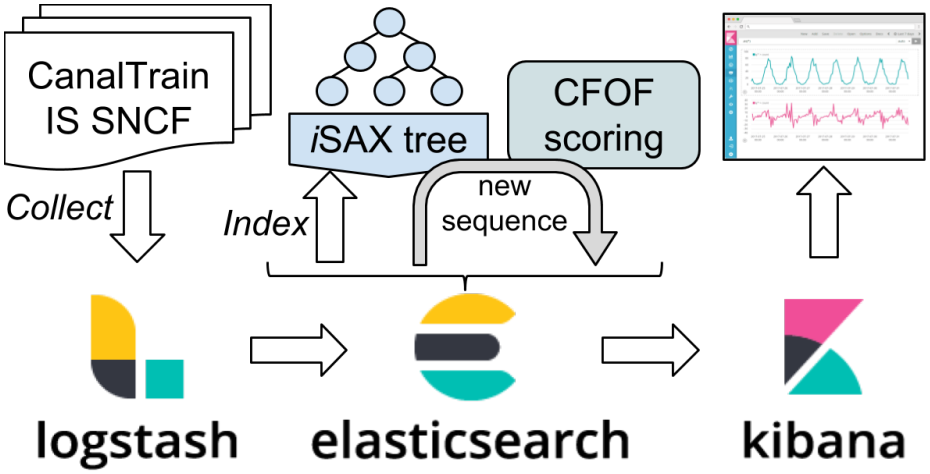

Fig. 2: Architecture of the system supporting the anomaly detection method.

The data preprocessing is structured in three steps: collection, aggregation and indexation. The data collection and aggregation steps are performed using requests to Elasticsearch, that return the result of the queries in JSON format, including the number of messages received per minute. Then, the data sequences are constructed by aggregating the measurements of the number of messages. The last preprocessing step consists in indexing past sequences in an iSAX tree, to build a history of objects of reference for the anomaly detection.

For every new sequence, an approximation of the CFOF anomaly score is computed with the method described in Section 3, and stored using Elasticsearch, in order to be available in Kibana for vizualisation.

5 Results

We present in this section the results of the evaluation of our anomaly detection method in the real industrial context of the SNCF communication infrastructure. The indicator that is monitored is the overall number of messages per minute.

To compute the sequences of reference (i.e., the objects stored in the iSAX tree) as defined in section 3, we use $dk = 24$ measures where each measure is the average over $dm = 15$ minutes of the number of messages per minute. Thus, each sequence contains 24 measures and represents 6 hours ($24 * 15$ minutes), which is consistent with the time scale of the railway activity regularities. One such sequence was built every 30 minutes (i.e., sequence shift $dd = 30$ minutes).

The same setting is used to build the new sequences for which the CFOF score is computed.

To illustrate the use of the method and of the system supporting it, we report results based on a period of reference starting the 1^{st} of November 2017 and ending the 31^{th} of August 2018. This period was preprocessed according to Section 3.1, leading to 14592 overlapping sequences. These sequences were extracted by requests to the Elasticsearch engine, and indexed into the iSAX tree.

5.1 Anomaly score

In this section, we present anomaly scores computed for the sequences from the 1^{st} of September 2018 to the 25^{th} of September 2018. These sequences were built in the same way as the sequences of reference. The measures used to construct the sequences over this period are shown in Figure 3 (top).

The approximation of the CFOF score obtained with our method for $\varrho = 0.001$ is given Figure 3 (bottom, in orange). On this figure, we can observe four periods of disturbance, that are pointed out by a higher CFOF score. The first occurs from the 6^{th} to 7^{th} of September, the second from the night between the 13^{th} and 14^{th} to the 15^{th} of September, the third during the 19^{th} of September, and the last spans from the evening of the 20^{th} to the 22^{th} of September. The analysis of these periods of disturbance is detailed Section 5.2.

The 14592 sequences of reference led to an iSAX tree containing 3449 nodes. The figure 4 shows that the pruning made by Algorithm 1 is effective and reduced the number of visited node during the approximation of the CFOF score. This computation took less than 2 minutes per sequence on a standard desktop computer (Intel Xeon Silver4114 at 3.40GHz with 4GB of main memory), and was 25 times faster than the computation of the exact score. This approximation can thus easily be performed on the fly, allowing to monitor the message stream with a sequence shift $dd = 30$ minutes.

Figure 3 (bottom) also shows the exact CFOF score (in blue). One can observe that the approximation closely follows the exact score. The Spearman rank-order correlation confirm the quality of the approximation. Its value, calculated between the real and estimated CFOF scores during the period reported in Figure 3, is 0.788. Computing the CFOF score with other ϱ values shows

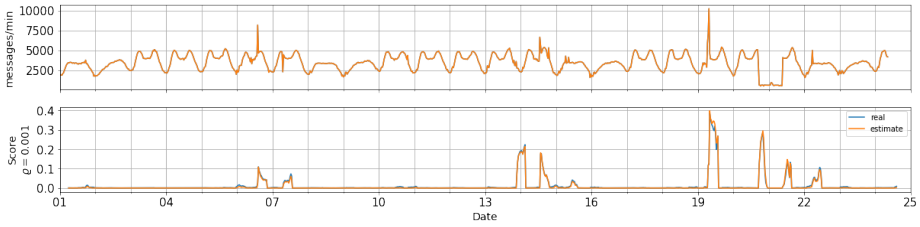

Fig. 3: Number of messages received per minute during the period from the 1$^{\text{st}}$ of September 2018 to the 25$^{\text{th}}$ of September 2018 (top). CFOF anomaly score over this period with $\varrho = 0.001$ (bottom).

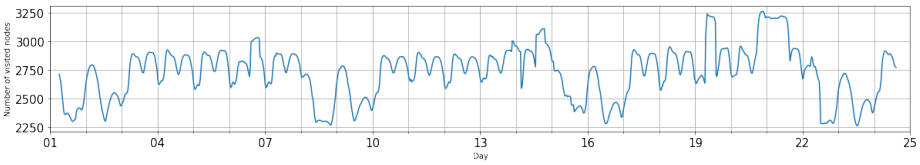

Fig. 4: Number of visited nodes in the iSAX tree during the computation of the CFOF approximation.

that when rho increases, the Spearman rank-order correlation is even better and increases monotonically. We observe for instance a value of 0.886 for $\varrho = 0.01$ and up to 0.998 for $\varrho = 0.1$. The reduction of execution times as well as the quality of the approximation were also confirmed on the synthetic datasets used to evaluate CFOF itself in [3]. On these datasets, the approximation was performed more than 40 times faster than the computation of the exact score, and the Spearman rank-order correlation ranged from 0.876 to 0.999.

To study the impact of the parameter ϱ on the detection, we consider two extreme values of ϱ. For $\varrho = 0.1$, the CFOF value of a sequence is computed by taking into account its rank in a neighbourhood of 10% of the sequences of reference. This leads to higher scores, that are based on a very stringent similarity requirement. The second value of $\varrho = 0.001$ takes into account the rank in a neighbourhood of 0.1% of the sequences of reference. This is a much weaker requirement, leading to lower scores. Some anomalies are not detected for all values of ϱ. For instance, in Figure 5, with $\varrho = 0.1$, we can observe two small anomalies the 17$^{\text{th}}$ and the 18$^{\text{th}}$ of September, that are not captured with $\varrho = 0.001$. However, the most salient anomalies are detected for all ϱ in $[0.001; 0.1]$, as for example the one occurring on the 19$^{\text{th}}$ of September (see Figure 6 (bottom) for other intermediate values of ϱ). One can see that with $\varrho = 0.05$ and $\varrho = 0.1$, the complete period between the 20th and 21st is detected as anomalous, while with $\varrho = 0.001$ and $\varrho = 0.01$, only the beginning and end are flagged. This is explained by the fact that the whole period is a pattern less frequently observed in the past than the transitions that start and end it

considered independently. In the case of an automatically-controlled detection process, the last step of the anomaly detection can be performed by setting a threshold above which the score generates an alarm message. In this context, the tuning of ϱ can be made by increasing/decreasing ϱ to be more/less tolerant to variations of the shape of the signal with respect to the sequences of reference. In our case, we do not rely on a fully automated detection process, but on experts that inspect dashboards. Thus, it is not necessary to select a single value of ϱ, and the CFOF anomaly scores is plotted for several values of ϱ on the same graph, and presented to the experts, as for instance in Figure 6 (bottom).

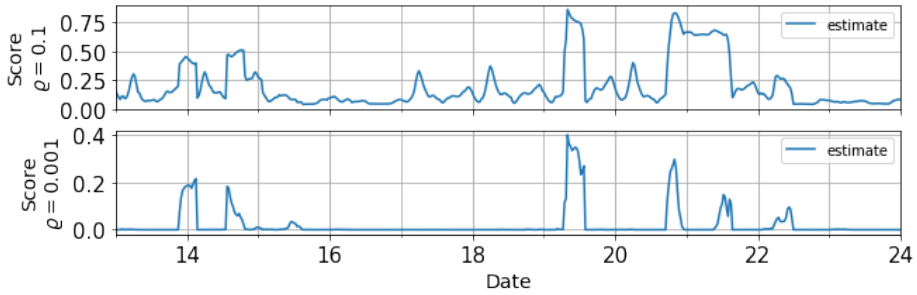

Fig. 5: Sensitivity of CFOF anomaly score with respect to ϱ.

5.2 Description of the anomalies

In this subsection, we describe the four periods of disturbance observed in figure 3. Within the SNCF information system, two main classes of known anomalies are related to messages buffering and signal collapse. The first kind of anomaly is caused by the near saturation at a given point in the processing and communication chain, entailing the buffering and accumulation of messages in a node of the information system. This creates oscillations in the signal as the buffer is emptied/filled. Such anomalies occur when a node reaches capacity limits or when a computing resource becomes too scarce. When this load bursts have been processed and corrective actions have been implemented, the traffic returns back to a normal state. The second class of anomalies occurs when the processing and communication pipeline becomes completely clogged at some point. This situation requires most of the time a partial or complete restart of the impacted equipement.

The first anomaly, from the 6^{th} to the 7^{th} of September, happened because of messages accumulated on an upstream platform. The period of disturbance, from the night between the 13^{th} and the 14^{th} to the 15^{th} of September has not been detected by neither the SNCF supervisors nor the different SNCF services. Our method however detected this anomaly, which could be interpreted

as the premises of the next one arising the 19$^{\text{th}}$ of September. The last anomaly, where the signal collapses during the night between the 20$^{\text{th}}$ and the 21$^{\text{th}}$ of September, results from an internal dysfunction of a message broker (Apache ActiveMQ). These two last anomalies (19$^{\text{th}}$ and 20$^{\text{th}}$-21$^{\text{th}}$ of September) are typical representatives of the two classes of known anomalies aforementioned (i.e., messages buffering and signal collapse).

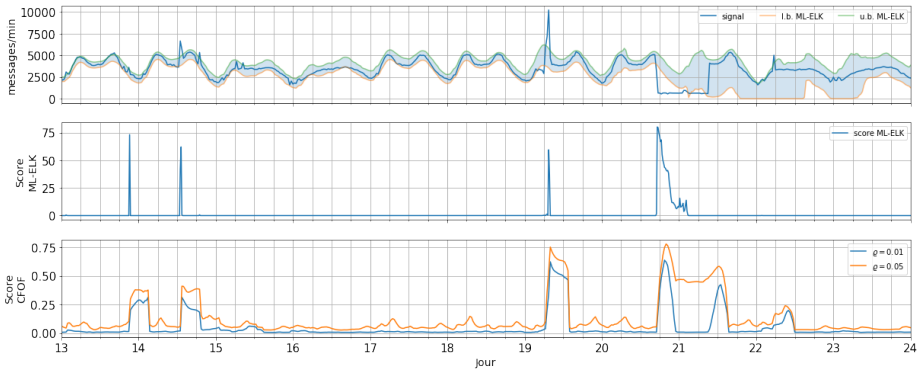

Fig. 6: Number of messaged received per minute during the period from 13th September 2018 to 24$^{\text{th}}$ of September 2018, and lower and upper bounds obtained by ELK-ML (top). ELK-ML anomaly score (middle) and CFOF anomaly score over this period for $\varrho = 0.01$ and $\varrho = 0.05$ (bottom).

5.3 Comparison with ELK Machine Learning method

In a second step of validation of the method, we compared the results with the anomaly detection method proposed by the ELK Machine Learning toolbox (ELK-ML) on the period ranging from the 1$^{\text{st}}$ of September to the 25$^{\text{th}}$ of September. We show on the figure 6 this comparison from the 13$^{\text{th}}$ to the 24$^{\text{th}}$ of September. The raw signal (the number of messages received per minute) is given Figure 6 (top) with the lower and upper bounds computed by the ELK-ML method. If the signal value stays between these two thresholds, then ELK-ML considers it as normal. Otherwise ELK-ML provides an anomaly score greater than zero as shown Figure 6 (middle).

When comparing these values with our anomaly scores given Figure 6 (bottom), it turns out that the anomaly on the 22$^{\text{nd}}$ of September is captured by our method but not by ELK-ML. For the other anomalies, both methods are able to underline the beginning of the anomaly. For ELK-ML the anomaly score quickly vanishes after the beginning of the anomaly (dropping to zero), whereas our method outputs a coherent score up to the end of the anomaly. Providing such information, about the anomaly durations, is very important in an indus-

trial context that requires to know if the communication system is back in a normal state or not.

6 Conclusion

In this paper, we focused on the problem of anomaly detection in the message streams of a railway communication system. We presented a method dedicated to this task, as well as a software architecture supporting it. This method is based on the collection and aggregation of message counts, the indexation of reference data, and the scoring of the new data arriving in the stream with respect to the reference data. The score used is the CFOF anomaly score, and we proposed an algorithm to efficiently compute an approximation of this score, using data indexing in an *i*SAX tree.

We presented experiments in a real industrial context, showing that the method is effective in detecting relevant anomalies. When compared to an approach of reference for this kind of data (the ELK Machine Learning toolbox), one of the main advantage of the method is the information it provides about the anomaly durations. The approximation of the score performed by the method is very close to the real CFOF score and is computed 25 times faster, enabling the scoring of the anomalies to be performed on the fly over the stream. Future work includes the application of the method to other indicators in the SNCF communication system, such as the variance of the message rate, or the average latency between two points of collect. Another promising direction is to use the context of the messages -as for example the type of the data or the type of the source- to define and compute context-dependant anomaly scores.

Acknowledgment

This work was performed within the framework of the LABEX IMU (ANR-10-LABX-0088) of Université de Lyon, within the program *Investissements dAvenir* (ANR-11-IDEX-0007) operated by the French National Research Agency (ANR). It was funded by the French National Association of Research and Technology (ANRT). We gratefully acknowledge strong support from the CNRS/IN2P3 Computing Center (Lyon/Villeurbanne - France), for providing a significant amount of the computing resources needed for this work.

References

1. D. M. Hawkins, *Identification of outliers*. Springer, 1980, vol. 11.
2. "ELK Stack Homepage," 2019-06-12. [Online]. Available: https://www.elastic.co/fr/elk-stack
3. F. Angiulli, "Concentration free outlier detection," in *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, 2017, pp. 3–19.

4. J. Shieh and E. Keogh, "iSAX: Indexing and mining terabyte sized time series," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2008, pp. 623–631.
5. ——, "iSAX: disk-aware mining and indexing of massive time series datasets," *Data Mining and Knowledge Discovery*, vol. 19, no. 1, pp. 24–57, Aug 2009.
6. R. A. Khan, S. Islam, and R. Biswas, "Automatic detection of defective rail anchors," in *Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems*, Oct 2014, pp. 1583–1588.
7. M. Molodova, Z. Li, A. Nez, and R. Dollevoet, "Monitoring the railway infrastructure: Detection of surface defects using wavelets," in *Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems*, Oct 2013, pp. 1316–1321.
8. L. Yin, J. Hu, L. Huang, F. Zhang, and P. Ren, "Detecting illegal pickups of intercity buses from their gps traces," in *Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Oct 2014, pp. 2162–2167.
9. M. D. Abràmoff, Y. Lou, A. Erginay, W. Clarida, R. Amelon, C. Folk, and M. Niemeijer, "Improved automated detection of diabetic retinopathy on a publicly available dataset through integration of deep learning," *Investigative ophthalmology & visual science*, vol. 57, no. 13, pp. 5200–5206, 2016.
10. V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 15:1–15:58, Jul 2009.
11. ——, "Anomaly detection for discrete sequences: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 823–839, May 2012.
12. K. M. Ting, F. T. Liu, and Z. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, 12 2008, pp. 413–422.
13. T. J. Veasey and S. J. Dodson, "Anomaly detection in application performance monitoring data," *International Journal of Machine Learning and Computing*, vol. 4, no. 2, p. 120, 2014.
14. S. Mukkamala, G. Janoski, and A. Sung, "Intrusion detection using neural networks and support vector machines," in *Proceedings of the 2002 International Joint Conference on Neural Networks*, vol. 2, May 2002, pp. 1702–1707.
15. X. Li, J. Han, S. Kim, and H. Gonzalez, "Roam: Rule- and motif-based anomaly detection in massive moving object data sets," in *Proceedings of the 2007 SIAM International Conference on Data Mining*, 2007, pp. 273–284.
16. C. C. Aggarwal, *Outlier Analysis*. Springer Publishing Company, Incorporated, 2013.
17. M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," *SIGMOD Rec.*, vol. 29, no. 2, pp. 93–104, May 2000.
18. E. M. Knox and R. T. Ng, "Algorithms for mining distance-based outliers in large datasets," in *Proceedings of the International Conference on Very Large Data Bases*. Morgan Kaufmann Publishers Inc., 1998, pp. 392–403.