

Anomaly Detection Using Similarity-based One-Class SVM for Network Traffic Characterization

Bouchra Lamrini¹, Augustin Gjini¹, Simon Daudin¹,
François Armando¹, Pascal Pratmarty¹ and Louise Travé-Massuyès²

¹LivingObjects, Toulouse, France

e-mail: {bouchra.lamrini,augustin.gjini,simon.daudin,françois.armando,pascal.pratmarty}@livingobjects.com

²LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France

e-mail: louise@laas.fr

Abstract

In this paper*, we investigate an unsupervised machine learning method based on one-class Support Vector Machines for anomaly detection in network traffic. In the absence of any prior expert knowledge on anomalous data, we propose the use of a similarity measure for Multivariate Time Series to evaluate the output results and select the best model. A set of Key Performance Indicators, oriented for network and traffic monitoring, has been used to demonstrate the promising performance of the unsupervised learning approach.

1 Introduction

Anomaly detection aims at identifying unusual patterns in data that do not conform to expected behavior [1]. These non-conforming patterns are generally referred in different application fields to as anomalies, aberrations, discordant observations, exceptions, novelty, outliers, peculiarities or contaminants, surprises, strangeness. There has been applications in several application fields from intrusion detection, e.g. identifying strange patterns in network traffic that could signal a hack [2], to system health monitoring, e.g. spotting a malignant tumor in an MRI image scan [3], and from fraud detection in credit card transactions [4], to fault detection in operating environments [5]. In this paper we are interested in anomaly detection in network traffic.

Support Vector Machines (SVMs) have been one of the most successful machine learning techniques that can be used in a variety of classification applications.

*Index Terms: Anomaly Detection, Support Vector Machines (SVMs), One-Class SVMs, Unsupervised Learning, Model Selection, Similarity Measure, Multivariate Time Series (MTS).

SVMs perform at least as good as other methods in terms of the generalization error [6].

Many factors contributed to the high popularity of SVMs today. First of all, their theoretical foundations have been deeply investigated and they come with a convex optimization procedure ensuring that the global optimum will be reached. Moreover, the solution is sparse making it really efficient in comparison to other kernel-based approaches [7]. In addition, they may use a non linear transformation in the form of a *kernel* that even allow SVMs to be considered as a dimensionality reduction technique [8]. One-Class SVMs [9] have been devised for cases in which one class only is known and the problem is to detect anything outside this class. This is known as novelty detection and it refers to automatic identification of unforeseen or abnormal phenomena [1; 10; 11], i.e. outliers, embedded in a large amount of normal data.

In contrast to traditional SVMs, One-Class SVMs (OC-SVM) learn a decision boundary that achieves maximum separation between the samples of the known class and the origin [12]. Only a small fraction of data points are allowed to lie on the other side of the decision boundary: those data points are considered as outliers.

Anomaly detection is particularly important for network traffic. The observed growth rate of informational and economic damage caused by intentionally or unintentionally attacks, faults, and anomalies has been a driving force behind efforts to ensure that network monitoring systems are able to detect abnormal activity and characterize it. The limitations of computing and storage resources require competence and ingenuity to effectively characterize ever-changing network traffic trends.

Non-availability of labelled data, high costs for constituting labeled training data, and need to identify anomalous and novel observations in data without having necessarily seen an example of that behaviour in the past are the main challenges tackled in this

work. A central issue is model selection, i.e. choice of the optimal hyper-parameters that define the OC-SVM learning configuration. This requires a method to evaluate the results.

This paper contributes to this problem by evaluating the results of the trained model by comparing samples predicted normal with samples in the training set. Because samples are composed of a set of signals over a temporal window, we propose to use a similarity index for Multivariate Time Series (MTS) called EROS (Extended Frobenius Norm). The results of the model are evaluated iteratively for different hyper-parameters of OC-SVM and the model that evaluates best is selected. We show that OC-SVM in combination with the Eros index [read more in Section 4.1] can create automatically tuned reliable classifiers with reasonable computation cost.

The remainder of this paper is organized as follows.

Section 2 presents our case study. Section 3 is devoted to an overview of SVMs and One-Class SVMs methods. In Section 4, we present the EROS similarity measure used to search for the best training model for anomaly detection. Experimental results and related discussions are provided in Section 5 to demonstrate the approach performance. Finally, Section 6 concludes the paper.

2 Case Study

Processing network traffic involves dealing with an immense amount of data that is quickly and constantly varying. Considering the enormous amount of data involved it is very easy for malicious activities to go undetected, especially without any knowledge a priori about the nature of the traffic, like it is often the case in the network domain.

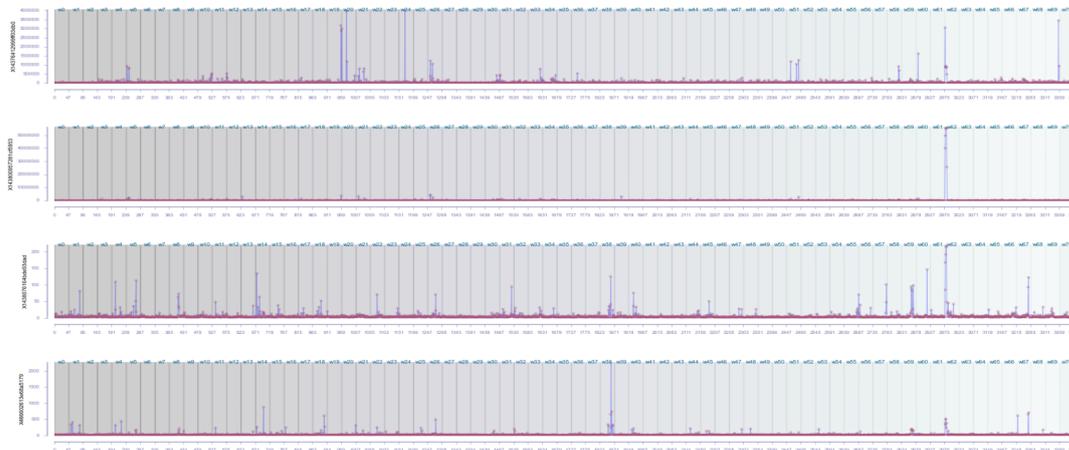


Figure 1: Training Data Set. The y-axis represents the KPI signals. From top to bottom: Total Incoming Traffic, Total Outgoing Traffic, Server Delay, and Network Delay.

In this study, data was collected from a real-time monitoring platform dedicated to ensure key application performance. 51 sites using applications of the same kind and having roughly the same uses at the same time, were chosen. For each application, we selected carefully four relevant Key Performance Indicators (KPIs) describing:

1. Total Incoming Traffic.
2. Total Outgoing Traffic.
3. Server Delay, i.e. the connection time to the server, which sets the expiration time for sending a request.
4. Network Delay that specifies how long it takes for a bit of data to travel across the network from one node to another.

A history of two months (3408 samples) of data generated at a 5 minutes rate was collected for each of the four KPIs above and for each site. Data was segmented into time-windows w_i , each of 48 points. Figure 1 shows the data contained in the training set. Each time-window $w_i \in \{w_0, \dots, w_l\}$, $l = 70$, is identified between lines and there are 70 samples.

Let us notice that the data samples that are provided to the OC-SVM classification algorithm are multivariate, each composed of four KPI segments over the same time-window. The idea is to detect insidious problems that require an analysis of the signal underlying interactions. We therefore want to detect “abnormal windows”. Each segment is characterized by seven statistical attributes: minimum (MIN), maximum (MAX), mean (MEAN), median (MED), stan-

dard deviation (STD), number of average crossings (nbpMean), squared mean error (SME) computed between the raw data and the linear fitting. These attributes are used by OC-SVM after normalization. Figure 2 shows the attributes built for the 70 segments of the four KPIs mentioned above. The seven at-

tributes MEAN, MED, MAX, MIN, STD, nbpMean, and SME are illustrated in sub-figures from left to right. We can already notice some overruns that will make considerable contribution to the classifier profile defined by the decision function.



Figure 2: Attributes built for training data KPIs. Each point is calculated on a segment of 48 points.

3 An Unsupervised Similarity-based Method for Anomaly Detection

Support Vector Machines (SVMs) have always been of interest in anomaly detection because of their ability to provide non-linear classification through a kernel function. Via this short overview, we show that SVMs are theoretically well founded. We briefly introduce the basic concepts of SVMs then focus on OC-SVM that we adopted in this study. A more detailed presentation can be found in [13] and a good example is available on URL using "LibSVM" library of Matlab.

3.1 Support Vector Machines

Let us consider the traditional two-class support vector machines in which we are given a set of n training instances $S = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$. $x_i \in R^d$, where y_i is the class label of the x_i instance and $y_i \in [-1, +1]$. The linear SVMs classifier recovers an optimal separating hyperplane maximizing the "margin" of the classifier with the equation: $w^T x + b = 0$, with $w \in F$ and $b \in R$ two parameters which determine the position of the decision hyperplane in feature space F (its orientation is tuned by w and its displacement by b). The decision function can thus be generally written as:

$$f(x; w, b) = \text{sign}(w^T x + b) \in \{-1, +1\} \quad (1)$$

where:

$$\text{sign}(w^T x + b) = \begin{cases} +1, & \text{if } (w^T x + b) \geq 0 \\ -1, & \text{otherwise} \end{cases}$$

The concept of SVMs is to find (w, b) such that the hyperplane is positioned at maximum distance of the nearest training samples of the two classes in order to reduce the generalization error. This distance defines the "margin". SVMs have first been proposed for linearly separable classification tasks. However they were extended to non-linearly separable classification problems. Some samples are allowed to violate the margin (soft-margin SVMs) and a non-linear decision boundary can be obtained by projecting the data into a higher dimension space thanks to a non-linear function $\Phi(x)$. Data points may not be linearly separable in their original space but they are "lifted" into a feature space F where a hyperplane can separate them. When that hyperplane is projected back into the input space, it has a non-linear shape. To prevent the SVM classifier from over-fitting noisy data, slack variables ξ are introduced to allow some data points to lie within the margin, and the parameter $C > 0$ (Eq.2) tunes the trade-off between the classification error on the training data and margin maximization. The objective function of SVM classifiers has the following minimization formulation:

$$\min_{w, b, \xi_i} \frac{\|w\|^2}{2} + C \sum_{i=1}^n \xi_i \quad (2)$$

Subject to:

$$\begin{aligned} y_i(w^T \phi(x_i) + b) &\geq 1 - \xi_i \\ \xi_i &\geq 0, i = 1, \dots, n \end{aligned}$$

The minimization problem is solved using Lagrange Multipliers $\alpha_i, i = 1, \dots, n$. The new decision function rule for a data point x is:

$$f(x) = \text{sign}\left(\sum_{i=1}^n \alpha_i y_i K(x, x_i) + b\right) \quad (3)$$

Every $\alpha_i > 0$ is weighted in the decision function and thus supports the machine. Since SVMs are considered to be sparse, there are relatively few Lagrange multipliers with a non-zero value.

The function $K(x, x_i) = \Phi(x)^T \Phi(x_i)$ is known as the *kernel function*. Since the outcome of the decision function only relies on the dot-product of the vectors in the feature space F (i.e. all the pairwise distances for the vectors), it is not necessary to perform an explicit projection. As long as a function K provides the same results, it can be used instead. This is known as the *kernel trick*.

Popular choices for the kernel function are linear, polynomial, and sigmoidal. In this study, we used the Gaussian Radial Base Function:

$$K(x, x_i) = \exp\left(-\frac{\|x - x_i\|^2}{2\sigma^2}\right) \quad (4)$$

where $\sigma \in R$ is a kernel parameter and $\|x - x_i\|$ is the dissimilarity measure. With this set of formulas and concepts we are able to classify a set of data point into two classes with a non-linear decision function.

The power of the method comes from using kernel functions, which enable it to operate in a high-dimensional, implicit feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space. This operation is often computationally cheaper than the explicit computation of the coordinates. Figure 3 illustrates a non linearly separable data set clustered by SVM with two different kernel functions: linear and radial based. The observations are plotted blue or magenta depending on the class and the background is darker as the distance from the hyperplane is higher. Scores are given on the right bottom corners and show a significant increase for the non linear kernel.

3.2 One-Class Support Vector Machines

One-Class SVMs (OC-SVMs) are used to separate the data of one specific class, the target class, from other data. They are trained with positive examples only, i.e. data points from the target class. There are two different approaches: the approach according to

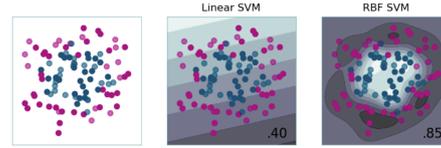


Figure 3: SVM results with two kernel functions.

Schölkopf [13], which is presented in the next paragraph, and that according to Tax and Duin [14].

In the feature space F , OC-SVM method basically separates all the data points from the origin by a hyperplane and it maximizes the distance of this hyperplane to the origin. This results in a binary function which captures the region of the input space where the training data lives. Thus the function returns $+1$ in a “small” region (capturing the training data points) and -1 elsewhere. The quadratic programming minimization function is slightly different from the original stated by (Eq.2) and (Eq.3):

$$\min_{w, \xi_i, \rho} \frac{\|w\|^2}{2} + \frac{1}{\eta n} \sum_{i=1}^n \xi_i - \rho \quad (5)$$

Subject to:

$$\begin{aligned} w \cdot \phi(x_i) &\geq \rho - \xi_i \\ \xi_i &\geq 0 \\ i &= 1, \dots, n \end{aligned}$$

Schölkopf et al. [13] has reformulated SVMs to take the new regularization parameter η instead of C in the original formulation (Eq.2 and Eq.3). The range of C is from zero to infinity, but η is always between $[0, 1]$. η characterizes the solution in a nice interpretable way: (1) it sets an upper bound on the fraction of outliers, e.g. the training examples regarded out-of-class, (2) and it sets a lower bound on the number of training examples used as support vectors.

Again by using Lagrange techniques and using a kernel function for the dot-product calculations, the decision function becomes:

$$\begin{aligned} f(x) &= \text{sign}((w \Phi(x_i)) - \rho) \\ &= \text{sign}\left(\sum_{i=1}^n \alpha_i K(x, x_i) - \rho\right) \end{aligned} \quad (6)$$

OC-SVMs thus create a hyperplane characterized by w and ρ which has maximal distance from the origin in the feature space F , hence separating all the data points from the origin.

4 Similarity-based Performance Evaluation for Model Selection

In this section, we address the problem of fitting the hyper-parameters of OC-SVM automatically, that is

the problem of automatic *model selection*. In the case of OC-SVM, this amounts to choose the kernel parameter γ and the regularization parameter η . A pair (γ_i, η_j) is defined as a learning configuration.

For this purpose, we propose to run OC-SVM for several learning configurations and select the best configuration by evaluating the similarity of the KPI signals for the windows tagged normal by OC-SVM and the KPI windows of the training data that are assumed to be normal examples. Since a sample window is composed of several KPI signals, we need a multidimensional similarity index for Multivariate Time Series (MTS).

4.1 The similarity Index Eros

Multidimensional similarity measures aim to indicate simultaneously the level of similarity between several datasets (databases, data clusters, etc.). Unlike other methods [15; 16; 17] that seek the level of similarity between two variables by omitting the existing correlation between the set of variables, a multidimensional method takes into account the contribution of each variable in defining a global similarity measure.

One of the methods processing MTS is the method *Eros (Extended Frobenius Norm)* [18]. The interest behind this method lies in its ability to assess the similarity of MTS composed of a different number of data points. It indeed uses the eigenvalues and eigenvectors of the covariance matrix that has size $n \times n$, n being the number of times series composing the MTS. In doing so, it also performs dimension reduction because the number of observations is generally higher than that of the variables.

We briefly describe the similarity index Eros based on the *Frobenius Norm* below. The definitions and notations used in this paper are taken from [19]. We first formally define the similarity index Eros. Next, we present the algorithm describing the similarity measure procedure and the approach proposed for model selection.

Definition 1. Eros (**E**xtended **F**robenius **N**orm). Let A and B be two MTS items of size $m_A \times n$ and $m_B \times n$ respectively. Let V_A and V_B two right eigenvector matrices by applying Singular Value Decomposition (SVD) to the covariance matrices, M_A and M_B , respectively. Let $V_A = [a_1, \dots, a_n]$ and $V_B = [b_1, \dots, b_n]$, where a_i and b_i are column-orthonormal of size n . The Eros similarity of A and B is then defined as:

$$\begin{aligned} Eros(A, B, w) &= \sum_{i=1}^n w_i |\langle a_i, b_i \rangle| \\ &= \sum_{i=1}^n |\cos(\theta_i)| \end{aligned} \quad (7)$$

where $\langle a_i, b_i \rangle$ is the inner product of a_i and b_i , w is a weight vector which is based on the eigenvalues of the MTS dataset, $\sum_{i=1}^n w_i = 1$ and $\cos(\theta_i)$ is the angle between a_i and b_i . The range of Eros is between 0 and 1, with 1 being the most similar.

Definition 2. Singular Value Decomposition. Let A be a general real $m \times n$ matrix. The singular value decomposition (SVD) of A is the factorization:

$$A = U\Sigma V \quad (8)$$

where U is a column-orthonormal $N \times r$ matrix, r is the rank of the matrix A , Σ is a diagonal $r \times r$ matrix of the eigenvalues γ_i of A where $\gamma_1 \geq \dots \geq \gamma_r \geq 0$ and V is a column-orthonormal $M \times r$ matrix. The eigenvalues and the corresponding eigenvectors are sorted in non-increasing order. V is called the right eigenvector matrix, and U the left eigenvector matrix.

Yang et al. (2005) [18] describe the similarity index algorithm with the following steps:

1. Compute the covariance matrix of each MTS.
2. Use SVD to decompose each covariance matrix.
3. Recover eigenvalues and eigenvectors.
4. Compute the weight w of individuals by normalizing the eigenvalues [18].
5. Compute similarity between MTS.

4.2 Automatic Model Selection

The first task is to define the learning configurations that will be tested with OC-SVM. We follow the steps below:

1. Define the hyper-parameter space and a procedure to explore this space. In our case, we set a min-max and a variation step to constitute a grid $(\beta \times \beta)$ value pairs, i.e. β values for each hyper-parameter.
2. Explore the hyper-parameter space and set OC-SVM accordingly: for each pair of values, one OC-SVM classifier is obtained after the learning step. The best configuration is retained by using the Eros similarity index on the validation data (25% of all data) and the training data (50% of all data). The corresponding OC-SVM classifier is taken as the best model.
3. Once found the best model, anomaly detection is performed on new data to evaluate how well the model behaves.

The similarity of windows tagged normal by OC-SVM, denoted by MTS_k^{normal} , $k = 0, \dots, p$, and the data windows of the training data (considered as normal), denoted by MTS_l^{learn} , $l = 1, \dots, q$, is obtained as follows. For every learning configuration [Figure 4] given by (γ_i, η_j) :

1. Compute Eros for every window pair $(MTS_k^{\text{normal}}, MTS_l^{\text{learn}})$, $k = 0, \dots, p$, and $l = 1, \dots, q$.
2. Compute the average similarity "Eros_{mean}" over

all the window pairs.

The best learning configuration is taken as the one leading to the maximal "Eros_{mean}" value over all considered learning configurations (γ_i, η_j) .

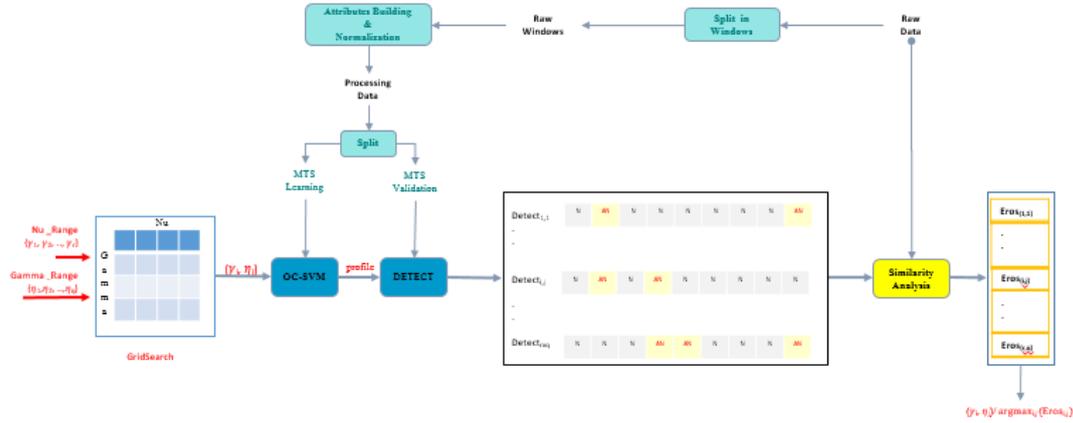


Figure 4: Diagram showing the model selection process.

5 Experiments on the Case Study

Our detection approach was applied to the case study presented in Section 2. A history of two months of data generated every 5 minutes for four KPIs was collected over 151 sites. The window segmentation [20] was performed after analyzing two points that can significantly impact the detection stage:

- choice of the time-window length, i.e. the number of hours and samples to take account in a window,
- definition of a reliable methodology to normalize training and testing datasets versus in these.

As mentioned in section 2, the time-window length was chosen of 4 hours, i.e. 48 samples. Clearly, when access to web applications is established in a few hours, a window of four hours is considered a significant period for traffic analysis. As noted above, each time-window is characterized by seven statistical attributes: minimum (MIN), maximum (MAX), mean (MEAN), median (MED), standard deviation (STD), number of average crossings (nbpMean), squared mean error (SME) computed between the raw data and their linear fit. The attributes are computed for each time-window in order to obtain a multidimensional scatter plot, where each point represents a time-window. One of the major interests of segmentation and feature computation is to synthesize the information contained in a time-window. This allows the de-

tection not only of singular points, but also of an atypical set of points even if each point taken.

Acquired raw data provide KPIs with different ranges, then features (attributes) themselves don't have homogeneous ranges. In order to guarantee good performances of the anomaly detection approach, we chose to normalize these attributes with respect to their maximal and minimal values with a tolerance using a threshold $s \in [0, 1]$. This standard preprocessing ensures that all the attributes contribute equally to the decision process independently of the parameters responsible of KPI dynamics.

To automatically select the best model, the hyperparameter space was discretized with a 10×10 grid, i.e. $\beta = 10$. 100 learning configurations were therefore evaluated to select the best model. This off-line task was performed for each application site and appeared computationally feasible.

Figure 5 shows some of the test results (25% of all data). From 24 time-windows ($w_i \in \{w_0, \dots, w_m\}$, $m = 23$), 4 anomalies have been detected represented by the 4 time-windows (yellow colored): w_3, w_4, w_{10} and w_{12} . The results were confirmed with Parallel Coordinates plots given in Figure 6.

In a Parallel Coordinates Plot, each attribute is given its own axis and all the axes are placed parallel to each other. Values are plotted as a series of lines that connect across all the axes. This means that each line corresponds to one data window for which we have 7×4 attributes (7 features for every KPI).

The order in which the axes are arranged can impact the way how the reader understands the data. One reason for this is that the relationships between adjacent variables are easier to perceive than those between non-adjacent variables. So re-ordering the axes can help in discovering patterns or correlations across variables. We clearly see that the four time-windows defined by the pink lines represent a strange behavior compared to the normal windows defined by the green lines.

Presenting this type of detection can ensure that the network administrators adopts another reasoning to characterize the nature of the traffic (normal, abnormal, critical, ...) circulating on the network. It may help him to identify the different forms of anomaly in his network. Data analysis must give meaning to the data with the goal of discovering useful information, suggesting conclusions, and supporting decision-making. The value of the data lies in the story it tells.

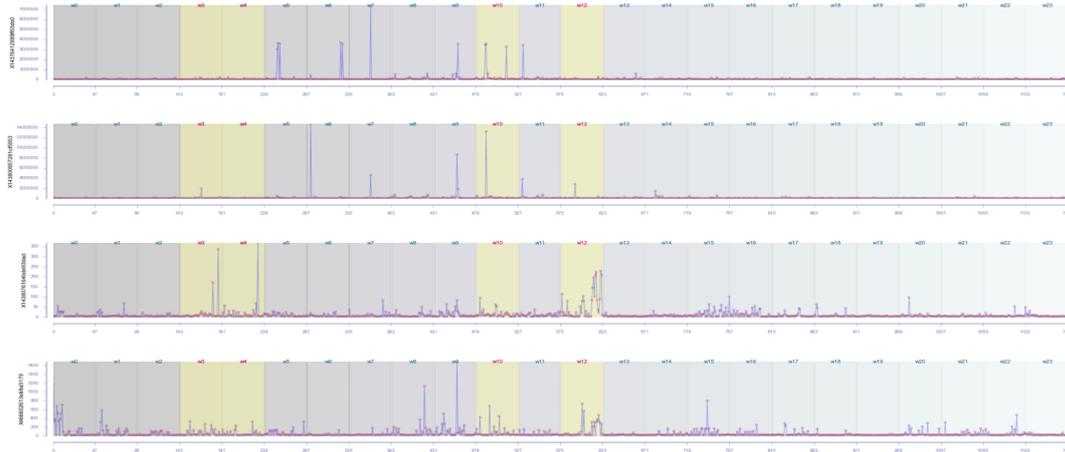


Figure 5: Anomaly detection (yellow windows are detected abnormal). From top to bottom, KPIs appear in this order on the y-axis: Total Incoming Traffic, Total Outgoing Traffic, Server Delay, and Network Delay

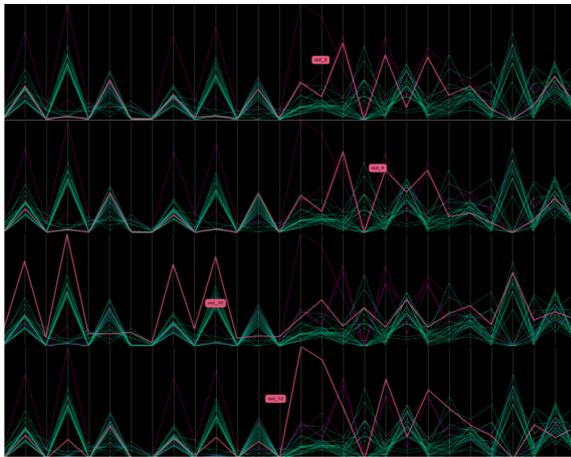


Figure 6: Parallel Coordinates Plot illustrating the four abnormal windows. From top to bottom, the curves labeled in pink color shows successively the time-windows: w_3 , w_4 , w_{10} and w_{12} .

6 Conclusion

In this work, we applied the OC-SVM method to detect anomalies in real network traffic, contributing with an automatic method based on the similarity index Eros [19] for setting the hyper-parameters which define the learning configuration. It provided very satisfactory results.

The advantages of novelty detection for complex processes like network traffic are multiple. In particular there is no need of faulty data. A wide variety of cases of anomaly exist and it would be impossible to characterize them all or to gather the corresponding data. Challenges for future work is related to the fact that data comes in a stream and dealing with the data in real-time is quite tedious. The amount of data leads to cases where resources are limited. Novelty detection in a distributed framework is also to be investigated.

Acknowledgement The authors thank Bertrand Le Marec and David Maisonneuve, leading team of LivingObjects, for their support and valuable comments about the application.

References

- [1] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15:1–15:58, 2009.
- [2] V. Kumar. Parallel and distributed computing for cybersecurity. *IEEE Distributed Systems Online*, 6(10):1–9, 2005.
- [3] C. Spence, L. Parra, and P. Sajda. Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model. In *Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA'01)*, MMBIA'01, pages 3–, 2001.
- [4] E. Aleskerov, B. Freisleben, and B. Rao. Card-watch: a neural network based database mining system for credit card fraud detection. In *Proceedings Of The IEEE/IAFE 1997 Computational Intelligence For Financial Engineering (CIFer)*, pages 220–226, 1997.
- [5] R. Fujimaki, T. Yairi, and K. Machida. An approach to spacecraft anomaly detection problem using kernel feature space. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, KDD'05*, pages 401–410, New York, NY, USA, 2005. ACM.
- [6] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.
- [7] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- [8] W. Wang, Z. Xu, W. Lu, and X. Zhang. Determination of the spread parameter in the gaussian kernel for classification and regression. *Neurocomputing*, 55(3-4):643–663, 2003.
- [9] M.A.F. Pimentel, D.A. Clifton, and L.C. Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.
- [10] D. Dasgupta and S. Forrest. Novelty detection in time series data using ideas from immunology. In *Proceedings of The 5th International Conference on Intelligent Systems*, Reno, Nevada, 1996.
- [11] E. Keogh, S. Lonardi, and W. Chiu. Finding surprising patterns in a time series database in linear time and space. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '02*, pages 550–556, New York, NY, USA, 2002. ACM.
- [12] B. Schölkopf, J.C. Platt, J.C. Shawe-Taylor, A.J. Smola, and R.C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.
- [13] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt. Support vector method for novelty detection. In *Proceeding of the 12th International Conference on Neural Information Processing Systems*, pages 582–588, 1999.
- [14] R.P.W. Tax, D.M.J. and Duin. Support vector data description. *Machine learning*, 54(1):45–66, 2004.
- [15] G.E.A.P.A. Batista, W. Wang, and E.J. Keogh. A complexity-invariant distance measure for time series. *SDM*, 2011.
- [16] C.A. Ratanamahatana and E.J. Keogh. Making time-series classification more accurate using learned constraints. In *Proceedings of SIAM International Conference on Data Mining (SDM'04)*, pages 11–22, 2004.
- [17] S. Park, W.W. Chu, J. Yoon, and C. Hsu. Efficient searches for similar subsequences of different lengths in sequence databases. In *16th International Conference on Data Engineering*, pages 23–32, 2000.
- [18] K. Yang and C. Shahabi. A multilevel distance based index structure for multivariate time series. In *12th International Symposium on Temporal Representation and Reasoning*, 2005.
- [19] K. Yang and C. Shahabi. A pca-based similarity measure for multivariate time series. In *Proceedings of the Second ACM International Workshop on multimedia Databases*, 2004.
- [20] S. Fuertes, G. Picart, J.Y. Tournet, L. Chaari, A. Ferrari, and C. Richard. Improving spacecraft health monitoring with automatic anomaly detection techniques. In *14th International Conference on Space Operations.*, page 2430, 2016.