SAD: An Unsupervised System for Subsequence Anomaly Detection

Paul Boniol, Michele Linardi, Federico Roncallo, Themis Palpanas
EDF R&D; LIPADE, Université de Paris
paul.boniol@edf.fr; {michele.linardi, federico.roncallo}@parisdescartes.fr; themis@mi.parisdescartes.fr

***Abstract*—Subsequence anomaly (or outlier) detection in long sequences is an important problem with applications in a wide range of domains. However, current approaches have severe limitations: they either require prior domain knowledge, or become cumbersome and expensive to use in situations with recurrent anomalies of the same type. We recently proposed NorM, a novel approach suitable for domain-agnostic anomaly detection, which addresses the aforementioned problems by detecting anomalies based on their (dis)similarity to a model that represents normal behavior. The experimental results on several real datasets demonstrate that the proposed approach outperforms the current state-of-the art in terms of both accuracy and execution time. In this demonstration, we present a system for unsupervised Subsequence Anomaly Detection (SAD) that uses the NorM method. Through various scenarios with real datasets, we showcase the challenges of the problem, and we demonstrate the advantages of the proposed system.**

I. Introduction

Massive data series[1] collections are a reality in virtually every scientific and social domain, and there is a pressing need for techniques that can efficiently analyze them [1]–[4].

Anomaly, or outlier detection is an old problem [5]–[7], finding applications in a wide range of domains. In the specific context of sequences, which is the focus of this paper, we are interested in identifying anomalous subsequences, that is, unlike the outlier, not a single value, but a sequence of values.

Existing techniques either explicitly look for a set of pre-determined types of anomalies [8], [9], or identify as anomalies the subsequences with the largest distances to their nearest neighbors (termed *discords*) [7], [10]. We observe that these approaches pose limitations to the subsequence anomaly identification task, for several reasons, explained below.

First, the anomalous behavior is not always known. Therefore, techniques that use specific domain knowledge for mining anomalies (e.g., in cardiology [8], and engineering [11]) involve several finely-tuned parameters, and do not generalize to new cases and domains. Second, in the case of general, techniques for subsequence anomaly detection, the state-of-the-art algorithms (e.g., [7], [10]) have been developed for the case of a *single* anomaly in the dataset, or multiple different (from one another) anomalies. The reason is that these algorithms are based on the distance of a subsequence to its Nearest-Neighbor (NN) in the dataset: the subsequence that has the farthest NN is marked as an anomaly. Figure 1 depicts this situation. We show a snippet of the MIT-BIH Supraventricular Arrhythmia Database (MBA) ECG recording [12] of patient *820*. This sequence includes *repeated* anomalous subsequences (*ventricular premature contractions*, marked by solid red rectangles). Following the state-of-the-art approaches [7], [10], we plot in Figure 1(b) the distance of each subsequence (of length *75*) to its NN, and we observe that the (known) anomalies do not correspond to the most distant NN (i.e., the highest peak in Figure 1(b)). This is because our dataset includes several anomalies that are similar to one another (i.e., of the same type). At the same time, these approaches mark as outliers subsequences that are normal (dotted black rectangle), resulting in false positives.

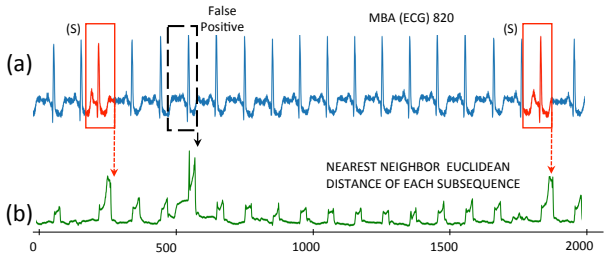

Fig. 1. (a) MBA ECG (*2000* points snippet from patient 820), with two anomalous Supraventricular premature beats (S). (b) Euclidean distances of each subsequence (length *75*) to its best non-trivial match in the full sequence: anomalies do *not* have the largest distance to their nearest neighbors.

Third, the m^{th} *discord* approach [13], which was proposed to remedy this situation, takes into account the multiplicity, m, of the anomalous subsequences that are similar to one another, and marks as anomalies all the subsequences in the same group, by computing the m^{th} (instead of the 1^{st}) NNs for each subsequence. However, this approach assumes that we know the multiplicity m, which is not true in practice.

In order to address the aforementioned problems, we proposed NorM [14], an unsupervised approach for subsequence anomaly detection. Contrary to all previous approaches, NorM detects anomalies based on their (dis)similarity to a model that represents the normal (expected) behavior. NorM starts by carefully selecting some of the subsequences of the dataset. The selected set of subsequences are then used to build the normal behavior model (itself a sequence). This process is automated, with no user intervention, and is effective even when the dataset contains multiple anomalies. Finally, NorM detects anomalies by comparing candidate subsequences to

[1]If the dimension that imposes the ordering of the sequence is time then we talk about *time series*. In the rest of this paper, we will use the terms *sequence*, *data series*, and *time series* interchangeably.

this normal model. An extensive evaluation has shown that NorM is statistically significantly more accurate than the current state-of-the art, and up to orders of magnitude faster [14].

In this demonstration, we present the Subsequence Anomaly Detection (SAD) system that is based on NorM. It is a web application that enables users to visualize data series, execute NorM and change its internal parameters, as well as compare to other anomaly detection algorithms.

II. NorM Anomaly Detection Framework

A. Problem Formulation

We formulate an approach for subsequence anomaly detection based on the notion of normal (expected) behavior. The set of all subsequences of length ℓ in a given data series T is defined as: $\mathbb{T}_\ell = \{T_{i,\ell} | \forall i . 0 \leq i \leq |T| - \ell + 1\}$. We assume that $\mathbb{T}_\ell$ contains both normal and anomalous subsequences. We define normal behavior as follows:

Definition 1 (Normal Model, N_M): Given a data series T, N_M is a model that represents the normal (i.e., not anomalous) trends and patterns of T.

Subsequence anomalies can then be defined in a uniform way: anomalies are the subsequences that have the largest distances to N_M (or their distance is above a set threshold).

Definition 2 (Subsequence Anomaly): Given a data series T, the set $\mathbb{T}_\ell$ of all its subsequences of length ℓ, and the Normal Model N_M of T, the subsequence $T_{j,\ell} \in \mathbb{T}_\ell$ with a distance to N_M $d = min_{i \in [0, \ell_{N_M} - \ell]} \{dist(T_{j,\ell}, N_{M_{i,\ell}})\}$ is an anomaly if d is in the $Top\text{-}k$ largest distances among all subsequences in $\mathbb{T}_\ell$, or $d > \epsilon$, where $\epsilon \in \mathbb{R}_{>0}$ is a threshold.

Note that the only essential input parameter[2] is the length ℓ of the anomaly (which is also one of the inputs in all relevant algorithms in the literature [7], [10], [13], [15], [16]).

The definition of N_M allows several interpretations. As we summarize in the following subsection (and detail elsewhere [14]), in this work we choose to define N_M as a sequence that summarizes normality in T, by representing the average behavior of a set of (ideally only) normal sequences.Intuitively, N_M is the data series, which tries to minimize the sum of Z-normalized Euclidean distances between itself and some of the normal subsequences in T. Last but not least, we need to compute N_M in an unsupervised way, i.e., without having normal/abnormal labels for the subsequences in $\mathbb{T}_\ell$.

Observe that this definition of N_M implies the following challenge: even though N_M summarizes the normal behavior only, it is computed based on T, which may include (several) anomalies. In our work, we address this challenge by taking advantage of the fact that anomalies are a minority class.

B. NorM framework

We now briefly describe the NorM framework [14] (refer to Figure 2). NorM detects anomalies based on their distance

[2]Parameter k (or ϵ) is not essential, provided we can *rank* the anomalies. In practice, experts first examine the most anomalous pattern, and then move down in the list (there is no rigid threshold separating anomalous from non-anomalous behavior [5]); anomaly discovery processes operate in this way.

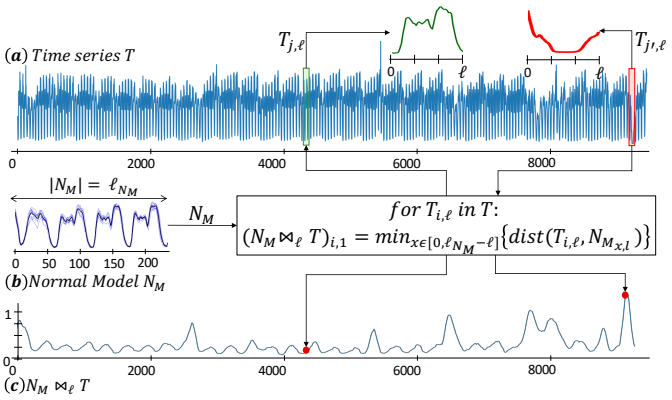

Fig. 2. NorM framework overview. The input time series (a) is used to build the Normal Model N_M (b). N_M is then used to compute the anomaly score $N_M \bowtie_\ell T$ (c). $T_{j',\ell}$ (red subsequence) is an anomaly (large distance to N_M), but $T_{j,\ell}$ (green subsequence) is not (small distance to N_M).

from the *Normal Model* sequence. It takes as input a data series T, and the length ℓ of the candidate anomalies. The algorithm first computes the Normal Model N_M based on T, and subsequently detects and returns a ranked list of the anomalous subsequences in T based on N_M. We note that the length of the anomalies, ℓ, is a user-defined parameter in all subsequence anomaly detection techniques, and can be set by the domain expert (e.g., in the case of electrocardiogram data, cardiologists are interested in analyzing heartbeats, which have a known length). The length of the Normal Model, ℓ_{N_M}, is automatically set to value larger than ℓ.

[Computing the Normal Model] Recall that N_M should capture (summarize) the normal behavior of the data. This may not be very hard to do for a sequence T that does *not* contain any anomalous subsequences. In practice however, we would like to apply the NorM approach in an unsupervised way on any sequence, which may contain several anomalies.

We compute the N_M sequence in three steps. First, we extract the subsequences that can serve as candidates for building the N_M. These candidates are either randomly selected from T (NorM-smpl), or correspond to motifs[3] (NorM-SJ). Then, we group these subsequences according to their similarity in a set of clusters $\mathbb{C}$ (we use hierarchical clustering and Minimum Description Length to identify the right number of clusters). The last step consists of scoring each cluster, and selecting the cluster that best represents normal behavior. Formally, for a given cluster $c \in \mathbb{C}$, we select the cluster that maximizes the following formula: $Norm(c, \mathbb{C}) = \frac{Frequency(c)^2 \times Coverage(c)}{\sum_{x \in \mathbb{C}} dist(Center(c), Center(x))}$, where $Frequency(c)$ is the number of subsequences in c, and $Coverage(c)$ is the time interval between the first and the last occurrence of a subsequence in c. Based on the subsequences of the selected cluster, we build N_M by computing its centroid (mean subsequence), depicted in Figure 2(b).

[Normal Model Based Anomaly Detection] Intuitively, the anomalous subsequences of the long series T are the ones that are far away from N_M. NorM first considers

[3]Motifs of T are the subsequences with the smallest distance to each other.

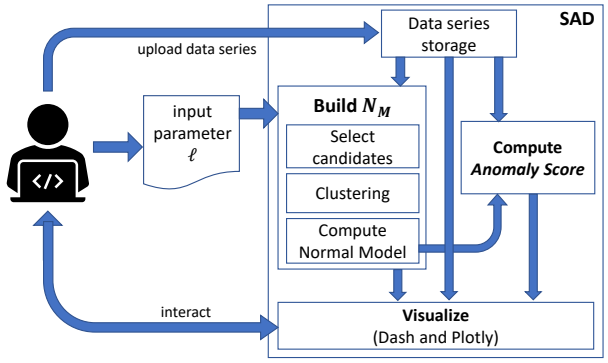

Fig. 3. SAD system architecture

the pairwise distances between each subsequence of length ℓ in T to subsequences of length ℓ in N_M, resulting in a meta-sequence, $N_M \bowtie_\ell T$, where $(N_M \bowtie_\ell T)_{i,1} = min(dist(T_{i,\ell}, N_{M,1,\ell}), ..., dist(T_{i,\ell}, N_{M,|N_M|-\ell+1,\ell}))$. $N_M \bowtie_\ell T$ (depicted in Figure 2(c)) contains at position i the nearest neighbor distance between subsequence $T_{i,\ell}$ and any subsequence of the same length (ℓ) in N_M. These distances correspond to the degree of abnormality: the larger the distance is to N_M, the more abnormal the subsequence is. We then extract the k subsequences of length ℓ with the highest distances in $N_M \bowtie_\ell T$, and rank them according to their distances. As such, $T_{j',\ell}$ (red subsequence in Figure 2(a)) is marked as an anomaly, but not $T_{j,\ell}$ (green subsequence) since its distance to N_M is small. Alternatively, we can extract all subsequences with distance larger than a threshold.

III. SAD Overview

The SAD GUI is an (upbeat!) stand alone web application, developed using Python 3.6 and the Dash framework. It enables users to load their own data series, T, and compute the normal model, N_M, and subsequently the anomaly score. The previous elements (T, N_M and anomaly score) are then inserted into a visualization frame, with which the user can interact. The overall architecture is shown in Figure 3. The front-screen of SAD is depicted in Figure 4.

[Interactive and Configurable Framework] The SAD GUI allows users to directly interact with the NorM framework and interface across the entire range of steps of the algorithm that executes under the hood. It first allows users to import their own datasets via an upload tab (top right button in Figure 4). If annotations (i.e., anomaly labels) are available, they can also be uploaded to SAD, which will use them in order to highlight the anomalous subsequences (in red) in the time series plot. The second functionality enables users to intervene and modify the operation of the NorM framework. SAD visualizes each step of the process (Subsequences selection, Clustering, Normal Model selection), and allows users to change the internal parameters of these steps of the algorithm. Figure 5 depicts this scenario. In Figure 5(left) we show the result of an execution of NorM; SAD displays the values for the internal parameters that were automatically selected by

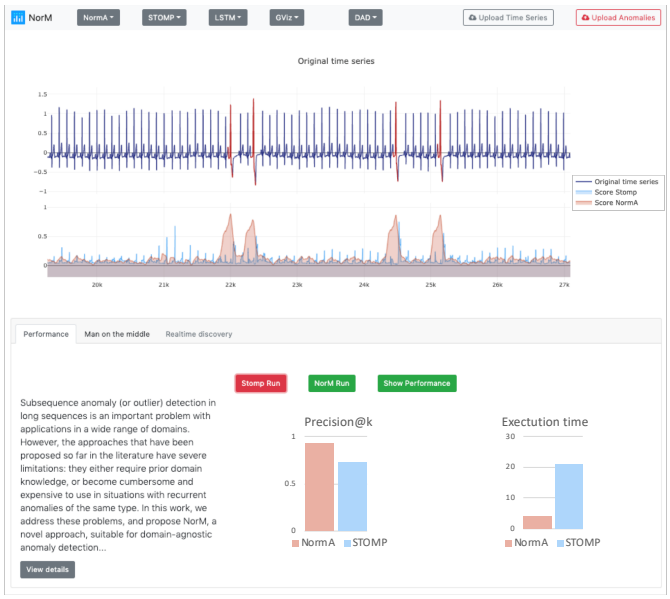

Fig. 4. Screenshot of SAD

NorM, along with N_M and the subsequences that were used to compute it. Figure 5(right) illustrates the result of a user manually selecting a (non-appropriate) cluster to be used for building N_M: we observe that the selected cluster contains very diverse subsequences, which lead to a non-representative N_M. Thus, SAD enables users to better understand how NorM works.

[Performance Visualization] SAD also supports comparisons to four state-of-the-art methods (i.e., STOMP [7], GrammarViz [10], DAD [13] and LSTM-AD [17]), by allowing users to execute multiple algorithms and superimpose their results in the GUI (see top half of Figure 4). Both the anomaly detection Top-k accuracy (if annotations were provided), and the execution time of all methods are visualized for easy comparison (e.g., see bottom right of Figure 4).

IV. Demonstration Scenarios

This demonstration has 3 goals: (i) show that SAD significantly speeds up the process of unsupervised subsequence anomaly detection; (ii) showcase the effectiveness of SAD and compare it to competing approaches, as well as to manual inspection; (iii) invite users to look inside the NorM framework, by visualizing each computational step and allowing them to change the internal parameters. In all cases, participants may select any of the available datasets, or upload their own.

[Scenario 1: Scalability] This exploration scenario begins with the long data series (100,000 points for approximately 50 anomalies) coming from the record 803 of the MIT-BIH Supraventricular Arrhythmia Database [12]. We will first run the state-of-the-art approaches, and display their time executions. We will then run NorM and compare the time performance. We thus demonstrate that NorM is significantly faster that previous state-of-the-art methods, and make it more suitable for long data series analysis.

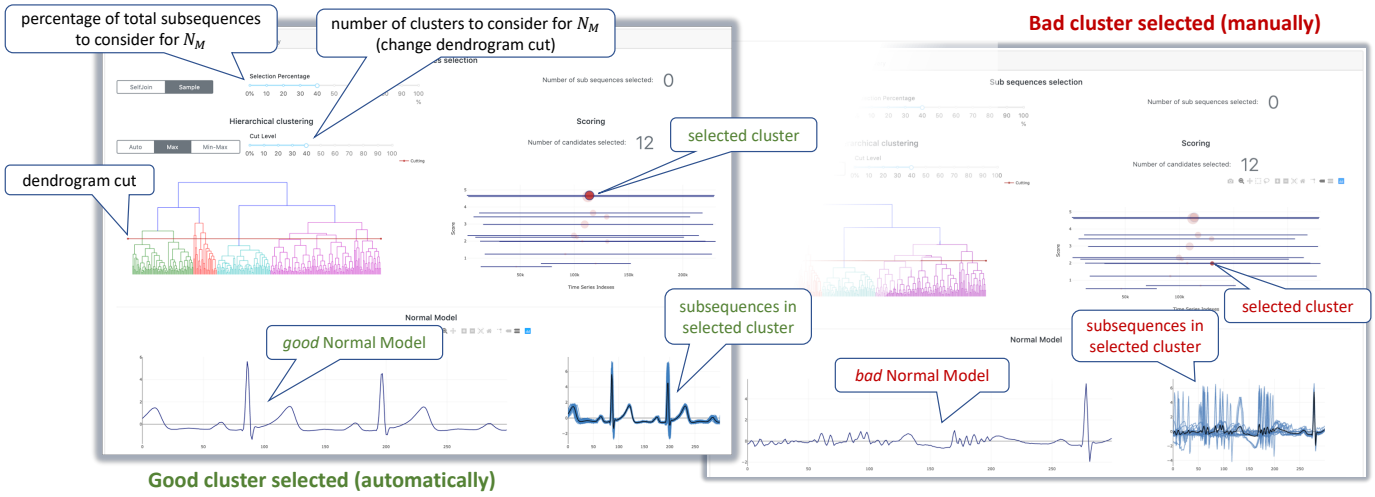

Fig. 5. Screenshots NorM steps highlighted by SAD: *(left)* NorM steps when the user picked the Normal Model suggested by SAD; *(right)* NorM steps when the user picked another cluster (bottom right plot).

[Scenario 2: Effectiveness] The second scenario will tackle the effectiveness (accuracy). We will start with two different data series coming from the MIT-BIH Supraventricular Arrhythmia Database (records 803 and 805, both with 100K points and 50-70 anomalies). We will run NorM and the competing approaches, and compare their Precision@k. Participants will observe that NorM outperforms the competitors, and discover that this is mainly due to the multiple similar anomalies that competitors cannot handle effectively.

[Scenario 3: Manual Anomaly Detection] The third scenario begins with two datasets. The first one corresponds to the New York City Taxi and Limousine Commissions dataset (NTC)[4] (10,000 points for 8 anomalies) and the record 820 of the MIT-BIH Supraventricular Arrhythmia Database (300,000 points for approximatively 100 anomalies). We will challenge participants to look for and identify anomalies in these datasets. The participants will be able to visualize the entire sequences, as well as zoom in/out and pan left/right. This exercise will help participants appreciate the difficulties and challenges of subsequence anomaly detection, especially when there are multiple anomalies, when these anomalies are subtle, and when the overall size of the sequence is large. Participants that perform well in the task, will receive a prize!

[Scenario 4: System Internals] The last scenario will allow the user to examine the way NorM works. It will expose to the user the inner-workings of the algorithm, and will allow them to change several internal parameters (number of subsequences selected as candidates, number of clusters, and which cluster to pick as the normal model). These changes will be applied in real time and will enable the user to understand how they affect the operation of NorM and the final results.

V. Conclusions

We propose SAD, a novel system, applicable to any domain, for subsequence anomaly detection that is based on the representation of normal behavior, which enables us to detect both single and recurrent anomalies, and leads to superior accuracy and time performance.

[4]http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

References

[1] T. Palpanas, "Data series management: The road to big sequence analytics," *SIGMOD Rec.*, vol. 44, no. 2, pp. 47–52, 2015.
[2] T. Palpanas and V. Beckmann, "Report on the first and second interdisciplinary time series analysis workshop (itisa)," *SIGMOD Rec.*, 2019.
[3] A. J. Bagnall, R. L. Cole, T. Palpanas, and K. Zoumpatianos, "Data Series Management (Dagst. Sem. 19282)," *Dagstuhl Reports*, 9(7), 2019.
[4] T. Palpanas, "Evolution of a Data Series Index," *CCIS*, 2020.
[5] V. Barnet, T. Lewis, *Outliers in Statistical Data*. J.Wiley & Sons, 1994.
[6] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using non-parametric models," in *VLDB*, 2006.
[7] C. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. Dau, D. Silva, A. Mueen, and E. Keogh, "Matrix profile I: all pairs similarity joins for time series: A unifying view that includes motifs, discords and shapelets," in *ICDM*, 2016.
[8] M. Hadjem, F. Naït-Abdesselam, and A. A. Khokhar, "St-segment and t-wave anomalies prediction in an ECG data using rusboost," in *Healthcom*, 2016.
[9] D. Abboud, M. Elbadaoui, W. Smith, and R. Randall, "Advanced bearing diagnostics: A comparative study of two powerful approaches," *MSSP*, vol. 114, 2019.
[10] P. Senin, J. Lin, X. Wang, T. Oates, S. Gandhi, A. P. Boedihardjo, C. Chen, and S. Frankenstein, "Time series anomaly discovery with grammar-based compression," in *EDBT*, 2015.
[11] J. Antoni and P. Borghesani, "A statistical methodology for the design of condition indicators," *Mechanical Systems and Signal Processing*, 2019.
[12] G. B. Moody and R. G. Mark, "The impact of the mit-bih arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, 2001.
[13] D. Yankov, E. Keogh, and U. Rebbapragada, "Disk aware discord discovery: Finding unusual time series in terabyte sized datasets," in *ICDM*, 2007.
[14] P. Boniol, M. Linardi, F. Roncallo, and T. Palpanas, "Automated anomaly detection in large sequence," in *ICDE*, 2020.
[15] Y. Liu, X. Chen, and F. Wang, "Efficient Detection of Discords for Time Series Stream," *Advances in Data and Web Management*, 2009.
[16] W. Luo and M. Gallagher, "Faster and parameter-free discord search in quasi-periodic time series," in *PAKDD*, 2011.
[17] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *ESANN*, 2015.