Real-Time Anomaly Detection for Advanced Manufacturing: Improving on Twitter's State of the Art

Caitríona M. Ryan, Andrew C. Parnell, and Catherine Mahoney

ABSTRACT

The detection of anomalies in real time is paramount to maintain performance and efficiency across a wide range of applications including web services and smart manufacturing. This paper presents a novel algorithm to detect anomalies in streaming time series data via statistical learning. Using time series decomposition, a sliding window approach and recursive updates of the test statistics, the generalised extreme studentised deviate (ESD) test is made feasible for streaming time series data. The method is statistically principled and it outperforms the `AnomalyDetection` software package, recently released by Twitter Inc. (Twitter) and used by multiple teams at Twitter as their state of the art on a daily basis. The methodology is demonstrated using unlabelled data from the Twitter `AnomalyDetection` GitHub repository, a manufacturing example with labelled anomalies and the Yahoo EGADS benchmark data.

1. Introduction

The detection of anomalies (data that deviates from what is expected) is important to protect revenue, reputation and resources in many applications such as web services, smart manufacturing, telecommunications, fraud detection and biosurveillance. For example, exogenic factors such as bots, spams and sporting events can affect web services as can hardware problems and other endogenic factors [7]. In advanced manufacturing, the detection of anomalies in streaming machine data, for example from machine sensors that monitor processing conditions, can aid the identification of tool wear and tear and any problems in the structure or quality of a part in production [17]. Figure 1 displays one such time series with labelled anomalies taken from the Numenta data repository [15]; temperature sensor data from an internal component of a large, industrial machine. The first anomaly is not explained or discussed in [15]. The second anomaly is a planned shutdown of the machine. The third anomaly is difficult to detect and directly led to the fourth anomaly, a catastrophic failure of the machine.

E-mail: triona.ryan@mu.ie

C.M. Ryan and A.C. Parnell are with I-Form / Hamilton Institute, Maynooth University, Ireland

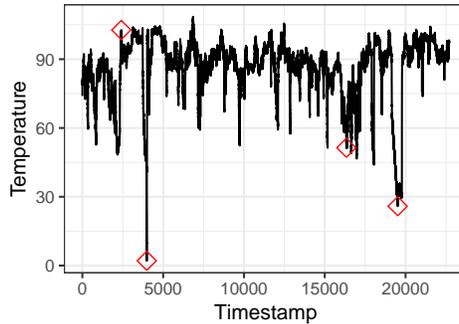C. Mahoney is with University College Dublin, Ireland.

Figure 1. Machine Temperature data stream with known anomalies marked in red

The `AnomalyDetection` software package [21] was recently released by Twitter and is used daily to detect anomalies in their cloud infrastructure data, for example Tweets Per Second (TPS) and CPU utilisation. A conference publication [22], an article published on ArXiv [7] and a blogpost [11] have generated much interest with over 120 citations since 2014, accepting Twitter's challenge to the public and academic community to "evolve the package and learn from it as they have" [11].

The problem of anomaly detection in time series data of this nature is challenging due to its seasonal nature and its tendency to exhibit a trend. The approach taken by Twitter [22, 7, 11, 21] is the Seasonal Hybrid Extreme Studentised Deviate (SH-ESD) algorithm. This is an adaptation of the generalised extreme studentised deviate (ESD) test [20] which is itself a repeated application of the Grubbs hypothesis test [5] for a single outlier. These tests assume that the data is normally distributed. Thus it is necessary to decompose the time series, subtract the seasonal and trend components and perform the hypothesis tests on the resulting residuals. In order to perform the decomposition, SH-ESD [22, 7, 11] uses the median value of non-overlapping windows of data to estimate the trend as a stepwise function, which they argue is more robust to outliers. Whilst this is computationally fast and works well for some datasets, the results can be especially sensitive to the choice of window size and location. SH-ESD uses LOESS to determine the seasonal component via `stl` [4]. However, SH-ESD requires the period to be specified by the user and a further requirement is that each non-overlapping window is assumed to capture at least one period of each seasonal pattern. In contrast, the approach presented in this paper avoids these restrictions and is applied to a rolling window of streaming data.

A major statistical consideration arises from the implementation of SH-ESD. In order to increase the robustness of the method against a large number of outliers, SH-ESD uses the median and median absolute deviations (MAD) to studentise observations. This is not statistically appropriate for the generalised ESD test. The ESD test is derived under a Central Limit Theorem assumption of normalisation by the mean and standard deviation. The resulting critical value approximation due to [19] accounts for the presence of outliers and thus does not require robust statistics such as median and MAD. The use of median and MAD for studentising observations may result in values that follow a heavier-tailed distribution than the adjusted t-distribution used in the significance tests. This is a potential cause of high levels of type I error when using SH-ESD.

This paper presents a novel algorithm entitled Recursive ESD (R-ESD) for fast anomaly detection in time series streaming data. It takes a statistically principled approach and addresses the problems with SH-ESD outlined above. Formulating the

2

test statistic in a novel recursive way allows the test to be implemented while streaming data in real time. This is a key improvement over existing methods, enabling a real-time, low memory, statistically principled version of a widely used approach. First, the seasonality and trend is estimated in an initial phase. Then the statistically principled generalised ESD test is implemented in a sliding window of time series data. The R-ESD approach results in fast identification of anomalies in each window which can be communicated to the end user while the data is being streamed. We address the problems in SH-ESD outlined above by; (i) formulating two recursive updates of the ESD test statistic, enabling anomaly detection while streaming in real time; (ii) using the mean to studentise observations, as it is statistically principled and appropriate for the statistical hypothesis test for outliers and (iii) estimating the period in the initial phase of the algorithm using a Fourier transformation via the `periodogram` function in TSA [2].

R-ESD is statistically principled. Moreover, SH-ESD has a number of further limitations where our method adds real value; (i) SH-ESD use non-overlapping windows of data. Thus its nature is prohibitive for true streaming data. R-ESD detects anomalies while streaming; (ii) In SH-ESD, each non-overlapping window is assumed to capture at least one period of each seasonal pattern. R-ESD is fast and easy to apply to a new dataset with no prior knowledge; (iii) In SH-ESD, this period must be pre-specified. This is not so for R-ESD; (iv) SH-ESD is prone to high levels of type one error (the detection of false positive anomalies), perhaps due to the incorrect use of median and MAD as described above. R-ESD outperforms SH-ESD and is statistically principled.

Other approaches to the problem of anomaly detection in times series data include Yahoo Inc.'s anomaly detection system; EGADS: the Extensible Generic Anomaly Detection System [14], DeepAnT [18] and DeepAD [1]. The latter two are acronyms of Deep Learning based Anomaly Detection methods. The strength of EGADS is in combining many forecasting models and anomaly detection methods to detect as many different types of anomalies as possible. As [14] admits, there is no best anomaly detection model for all use cases and certain algorithms are best at detecting different types of anomalies. Using EGADS, SH-ESD performed best on A3 (also known as TS-3) (see Figure 5 of [14]), which includes outliers but no change points. Section 4 demonstrates that R-ESD outperforms SH-ESD, thus it also outperforms EGADS for this dataset. DeepAnT (Munir et al., 2018) and DeepAD (Buda et al., 2018) are threshold based whereas R-ESD uses a principled statistical hypothesis test to discern an anomaly. DeepAnT uses convolutional neural network models (CNN) to predict the next value of a rolling window of time series data. A threshold on the Euclidean distance from actual to predicted value is used for the anomaly detection. DeepAD combines the predictions of multiple forecasting techniques to forecast each newly streamed datapoint and detect anomalies using a dynamic extreme value threshold. In both DeepAnT and DeepAD, one or multiple models are fit at every newly streamed datapoint whereas in R-ESD one model is fit for the entire dataset. In this sense the most comparable method to R-ESD is DeepADVote, a subset of DeepAD that fits only one model. This article demonstrates that R-ESD outperforms DeepADVote in F1-Score for the Yahoo EGADS Webscope benchmark A3 data [13]. Further comparisons and discussion of the benefits of R-ESD over and above its competitors will be demonstrated in Section 4.

The paper is organised as follows. Section 2 describes the problem of anomaly detection, including a short review of existing methods and outlines how we propose to measure the performance of R-ESD. Section 3 formally defines the problem and presents the R-ESD method. In Section 4, the R-ESD approach is demonstrated using

3

the manufacturing dataset displayed in Figure 1, using unlabelled data from the Twitter `AnomalyDetection` GitHub repository [21] and using the Yahoo EGADS Webscope A3 benchmark data [13]. We conclude with a discussion in Section 5.

2. Anomaly detection

Anomaly detection is the problem of identifying patterns in data that do not conform to typical behaviour. By definition, the anomaly detection problem depends on the data and or application in question. [3] provide a thorough review and compare a range of approaches to anomaly detection given in the scientific literature including examples from industrial damage detection, medical anomaly detection, cyber intrusion detection and sensor networks. [6] provide a survey of anomaly detection methods in the computer science literature for temporal data. The challenge of selecting a suitable algorithm is discussed in [10]. [16] presents a framework to identify and compare suitable methods for a water-quality problem to detect anomalies in high frequency sensor data.

In this paper we focus on the problem of univariate time series streaming data such as those presented by Twitter [22, 7, 11] and the manufacturing problem displayed in Figure 1. This problem is of major importance, evidenced by the `anomalydetection` package, `tsoutliers`, `anomalize` and other highly cited packages being similarly focused. The typical behaviour of data of this nature is to exhibit trend and/or seasonality. The existence of a trend might itself be an anomaly. The research challenge is to detect anomalous data points as they arrive in streaming applications or soon after they arrive. The exact nature of the streaming capacity will be application dependent.

Consider a univariate time series data stream $\ldots, x_{t-1}, x_t, x_{t+1} \ldots$, where x_t is an observation recorded at time t. We implement a rolling window approach to the problem of anomaly detection, where the window $\mathbf{x} = \{x_{t-w}, x_{t-w+1}, \ldots, x_t\}$ is the set of the w observations of the data stream prior to and including time t. The goal is to detect anomalies $\tilde{\mathbf{x}} \in \mathbf{x}$ and $\tilde{\mathbf{t}} \in \{t-w, \ldots, t\}$, the anomalies and associated time-stamps of a subset of observations in each rolling window while streaming.

In order to assess an anomaly detection algorithm, it is useful to analyse datasets that are annotated with ground truth labels. In such cases, one can measure the precision; the proportion of true positive anomalies of all detected anomalies and recall; the ratio of true positive anomalies to the sum of true positive anomalies and false negative anomalies. We use these measures to compare our performance to the Twitter `AnomalyDetection` package in detecting known anomalies in manufacturing data.

3. Recursive ESD for streaming time series data

The Grubb's test provides a hypothesis test for a single outlier [5]. This was generalised to the ESD test [20], where a pre-specified number of k anomalies can be detected. The ESD test statistics $R_1, \ldots, R_k$ are calculated from samples of size $n, n-1, \ldots, n-k+1$, successively reduced by the most extreme deviate (and potential anomaly) in the sample. For example, in the full sample of size n, the most extreme deviate would correspond to x_i, such that $\|x_i - \bar{x}\| \geq \|x_j - \bar{x}\| \quad \forall i, j = 1, \ldots, n$, with equality only when $i = j$. $\bar{x}$ is the full sample mean. This is computed analogously for subsequent reduced samples.

4

In general, we denote $\tilde{x}_j$ as the dataset with the jth most extreme deviates removed and $\tilde{n}_j$ as the sample size of this set. So, for example $\tilde{n}_1$ will be equal $n-1$ when the most extreme deviate is removed. The ESD test statistic is defined by

$$R_{j+1} = \frac{\max_i |x_i - \bar{\tilde{x}}_j|}{\tilde{s}_j}, \quad i = 1, \ldots, \tilde{n}_j; \quad j = 1, \ldots, k;$$

where the reduced sample mean is

$$\bar{\tilde{x}}_j = \frac{\sum_{i=1}^{\tilde{n}_j} \tilde{x}_i}{\tilde{n}_j},$$

and where the sum of squares with all the jth most extreme deviates removed is

$$\tilde{s}_j^2 = \frac{\sum_{i=1}^{\tilde{n}_j} (x_i - \bar{\tilde{x}}_j)^2}{\tilde{n}_j - 1}.$$

The critical values for this series of Student's t-tests are

$$\gamma_{l+1} = \frac{t_{n-l-2,p}(n-l-1)}{\sqrt{(n-l-2+t_{n-l-2,p}^2)(n-l)}}, \quad l = 0, 1, \ldots, k-1, \tag{1}$$

where n is the number of data points in the dataset, k is the maximum number of anomalies, l is the order statistic and $p = 1 - (\alpha/2)(n-l)$. Further details can be found in Equation 2.5 of [20].

In order to adopt the ESD test for streaming data, we note that the ESD test statistic R_{j+1} can also be expressed as a function of the Grubb's ratio $\frac{S_n^2}{S^2}$ used in [5] such that, in our notation;

$$R_{j+1} = \sqrt{\left(1 - \frac{\tilde{S}_{j+1}^2}{\tilde{S}_j^2}\right)(\tilde{n}_j - 1)}, \tag{2}$$

where

$$\frac{\tilde{S}_{j+1}^2}{\tilde{S}_j^2} = \frac{\sum_{i=1}^{\tilde{n}_{j+1}} (x_i - \bar{\tilde{x}}_{j+1})^2}{\sum_{i=1}^{\tilde{n}_j} (x_i - \bar{\tilde{x}}_j)^2}.$$

This construction of the ESD test statistic is novel and useful in the context of our streaming anomaly detection problem as it permits recursive calculations. Having identified x^* as the most extreme deviate in a sample $\tilde{x}$, the sums of squares can be reduced using the following recursive calculation;

$$\tilde{S}_{j+1}^2 = \tilde{S}_j^2 - \tilde{n}_{j+1}(x^* - \bar{\tilde{x}}_j)^2/\tilde{n}_j. \tag{3}$$

The recursive ESD test is outlined in Algorithm 1. It enables anomaly detection in streaming data by using a rolling window approach and recursively updating the test statistics recursively as each data point arrives. Let x_w be a newly streamed data point

---
Algorithm 1 Recursive Extreme Studentised Deviate test

Inputs: Dataset $\mathbf{x} = (x_1, x_2, \ldots, x_n)$
Significance level α;
Maximum number of anomalies to be tested k;
The initial full sample mean $\bar{\bar{x}}_0 = \bar{x}$ and $\tilde{S}_0^2 = \sum_{i=1}^{n} (x_i - \bar{x})^2$;

Algorithm:
for $j = 1$ **to** k **do**
    Identify x^*, the maximum deviate in the dataset;
    Perform a recursive update of the sum of squares using Equation 3;
    Calculate the critical value γ_{j-1} and test statistic R_j using Equations 1 and 2;
    if $R_j > \gamma_{j-1}$, **then**
        Flag x^* as an anomaly and attach to an anomaly vector $\mathbf{x}_A$;
        Recursively update $\bar{\bar{x}}_j$, the mean of the reduced dataset;
        Reduce the dataset by removing x^*;
    end if
end for

Outputs: Anomaly vector $\mathbf{x}_A$.
---

and let x_0 be the datapoint that is being removed as the window rolls forward by one at time t. Then the sum of squares and the sample mean can be calculated at time $t + 1$ by the following recursive formulae;

$$
\begin{aligned}
S_{t+1}^2 &= S_t^2 + (x_w - x_0)\left(x_w + x_0 - 2\bar{x}_t - \frac{x_w - x_0}{w}\right); \\
\bar{x}_{t+1} &= \bar{x}_t + \frac{(x_w - x_0)}{w}.
\end{aligned}
\tag{4}
$$

The Recursive ESD (R-ESD) streaming algorithm for anomaly detection is presented in Algorithm 2. It is a two stage approach. In the initial phase, a window of data $\mathbf{x}'$, of size w', is decomposed into its seasonal ($\mathbf{S}$), trend ($\mathbf{T}$) and residual ($\boldsymbol{\epsilon}$) components, such that

$$
x_t' = S_t + T_t + \epsilon_t,
$$

where $\epsilon_t \sim N(0, \sigma^2)$, that is, the residuals at each time step are assumed normally distributed with zero mean and variance σ^2 to be estimated. We also assume that the errors in this general model are uncorrelated in time. These assumptions render the generalised ESD test appropriate to detect anomalies in the residuals $\boldsymbol{\epsilon}$. Note that the initial window size $w' \geq w$ to allow the fit of a useful statistical model. We assume that in this training period, no anomalies are detected. In practice for example, an engineer would monitor a manufacturing process carefully during this initial phase.

Time series decomposition is a well-studied topic and commonly used methods are described in Chapter 6 of [8]. A suitable period p can be found in the initial phase for example, by using a Fourier transformation via `periodogram` from the `TSA` software package [2]. The `stlm` function of the `forecast` package [9] is then used to model the trend using LOESS and to forecast the typical behaviour of future observations as far as is required. For example, in the context of smart manufacturing, this would be the length of the production process that is about to be performed. If this is not computationally feasible or if the typical behaviour of the process is expected to

---
Algorithm 2 Recursive ESD Streaming Algorithm (R-ESD)

Inputs: Time series data $\mathbf{x} = (x_1, x_2, \ldots, x_t)$ observed at time t, with streaming new observations $(x_{t+1}, \ldots)$;
Initial window size w';
Streaming window size w;
Maximum number of anomalies k in any given window;

Initial Phase:
Define the initial training window of data by
$\mathbf{x}' = (x_{t-w'+1}, x_{t-w'+2}, \ldots, x_t)$;
Perform trend and seasonal decomposition of $\mathbf{x}'$ e.g. by using methods described in Section 3;
Create forecasts e.g. using the `forecast` function [9] $\mathbf{x}^f = (x_{t+1}^f, \ldots)$ as far as is required for the application and/or is computationally feasible;
Define the current window of data to search for anomalies by $\mathbf{x} = (x_{t-w+1}, x_{t-w+2}, \ldots, x_t)$ and denote the associated stationary residuals $\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2 \ldots, \epsilon_w)$ found in the model decomposition in line 2;
Compute the initial sum of squares of the residuals S_t^2 using Equation 5;
Compute the initial mean of the residuals $\bar{\epsilon}_t = \sum_{i=1}^{w} \epsilon_i / w$;

Streaming Phase:
for $s = (t+1)$ **to** $\ldots$ **do**
    Let $\epsilon_0 = \epsilon_1$;
    Slide the current window of data by one observation such that $\mathbf{x} = (x_{s-w+1}, x_{s-w+2}, \ldots, x_s)$;
    Calculate $\epsilon_w = x_s - x_s^f$ using the forecasts found in line 2;
    Update S_s^2 and $\bar{\epsilon}_s$ recursively (Equation 4);
    Perform the Recursive Extreme Studentised Deviate test to detect anomalies $\mathbf{x}_{A,s}$ (Algorithm 1);
end for

Outputs: $\{\mathbf{x}_{A,s}, t_s\}_{s=(t+1)}^{\cdots}$, where $\mathbf{x}_A, s$ is a vector of anomalies found using the R-ESD test in the sth window and where t_s denotes the time stamp of the last observation in the relevant window.
---

change across time, a suitable model can be re-fitted as often as is deemed necessary. The resulting model is used to calculate the residuals in the streaming window $\mathbf{x}$ and initialise the statistics required for the generalised ESD test namely, the mean of the residuals; $\bar{\epsilon}_t = \sum_{i=1}^{w} \epsilon_i / w$ and the sum of squares of the residuals;

$$S_t^2 = \sum_{i=1}^{w} (\epsilon_i - \bar{\epsilon}_t)^2. \tag{5}$$

In the streaming phase, the recursive generalised ESD test outlined in Algorithm 1 is applied to the residuals in each window. As the window slides forward by one datapoint at each iteration, fast recursive updates of S_t^2 and $\bar{\epsilon}_t$ at time t are employed using Equation 4 during streaming R-ESD (line 11, Algorithm 2).

4. Results

The R-ESD approach is first demonstrated and compared with SH-ESD using the "raw_data" example presented in the `AnomalyDetection` package published on the GitHub repository [21]. The anomalies are unlabelled but it is chosen for ease of comparison with SH-ESD. A single window of data is used to make as direct as possible a comparison and the full dataset is used while streaming. All other examples are for streaming data. The labelled machine temperature manufacturing problem introduced in Figure 1, Section 1 is useful to aid visualisation and understanding for a novel reader. Finally theYahoo EGADS benchmark data A3 (also known as TS-2) [13] is used to aid comparisons with EGADS and DeepAD. If no prior knowledge is available selecting $wprime = 10\%$ and $w = 2\%$ of the data is suggested with sensitivity analysis performed around these choices.

Figure 2 displays the R-ESD and SH-ESD results for a single window of 4 days of the first example. The significance level used for the generalised ESD test was $\alpha = 0.05$, the default in the `AnomalyDetection` package. The number of anomalies tested by R-ESD in the single window is $k = 288$. This is to coincide with max_anoms = 0.02 as is given in the `AnomalyDetection` package, the "maximum number of anomalies that S-H-ESD will detect as a percentage of the data" (since $k = w\times$ max_anoms $= 288$). R-ESD and SH-ESD agree in detecting 106 anomalies with a further 24 distinct anomalies detected by R-ESD and 8 by SH-ESD. The R-ESD anomalies appear to be more convincing although there is no ground truth for this example. CPU time for SH-ESD was 0.17 seconds, while R-ESD required 0.46 seconds. However, while the R-ESD is computationally less efficient than SH-ESD for this example, it is statistically principled and the algorithm is designed using recursive updates of the test statistic to allow real-time anomaly detection while streaming new datapoints using a sliding window.
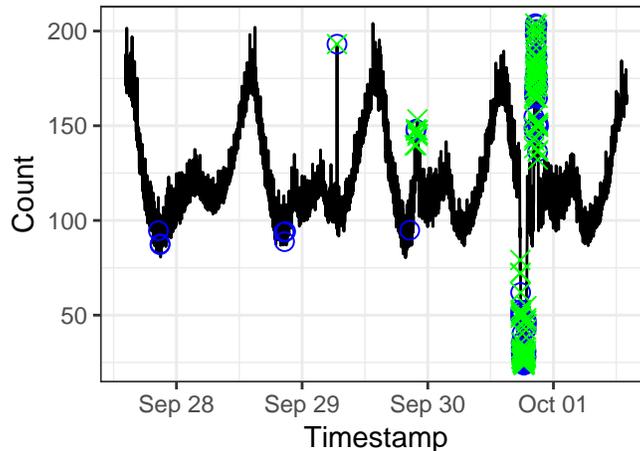

Figure 2. A single window of Twitter data: Resulting anomalies detected by R-ESD and SH-ESD are shown in green and blue respectively, with $k = 288$ (and equivalently max_anoms= 0.02) anomalies allowed in the window. A 4 day window is used from the example given in Twitter `AnomalyDetection` package on their GitHub repository with unlabelled anomalies.

Results for streaming windows of size $w = 1440$, that is, one day of minutely data, are given in Figure 3 and are compared to the non-overlapping window ap-

proach of SH-ESD described in [22] using the `longterm_period=TRUE` option in the `AnomalyDetection` package. As described in Section 2, the trend in each non-overlapping window is treated by SH-ESD as a flat line corresponding to the median of the values of the data in the window. Therefore, one might expect that R-ESD is less sensitive than SH-ESD to the choice of window size and certainly to the starting point of the algorithm. The CPU time for R-ESD to stream 7197 data points in this example was 65 seconds, that is only 0.02 seconds per window, where the window size was $w = 1440$. Thus streaming is highly feasible for many applications. Moreover, the R-ESD streaming approach seems to choose more sensible anomalies than the non-overlapping window approach of SH-ESD. Agreement occurs for 39 anomalies with a further 118 and 103 anomalies detected by R-ESD and SH-ESD respectively.
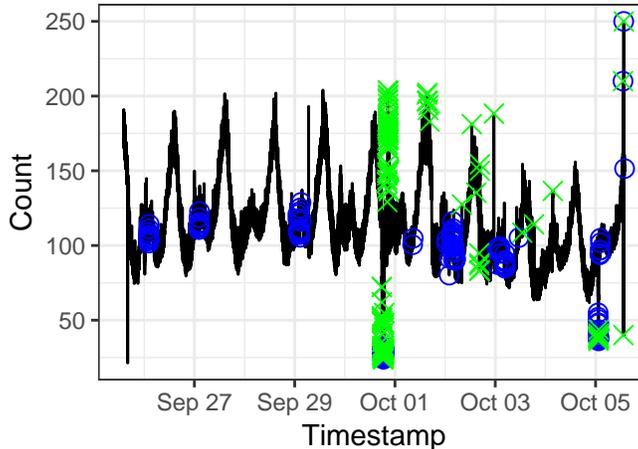

Figure 3. Streaming Twitter data: Resulting anomalies detected by R-ESD and SH-ESD are shown in green and blue respectively. The number of anomalies tested per window is $k = 288$ (and equivalently max_anoms= 0.02 for SH-ESD) anomalies allowed in the window.

The improved performance of R-ESD over SH-ESD is further demonstrated in the machine temperature example displayed in Figure 4 for the data described in Section 1. Here the number of anomalies per window $k = 10$ was deemed appropriate in the context of manufacturing such that parts are produced within the desired specification. There are 4 known anomalies in this dataset and these are noted as being difficult to detect, the third in particular [15]. SH-ESD failed to detect any of the 4 known anomalies when using the non-overlapping window approach to the anomaly detection. R-ESD performs better, providing anomaly detection in advance of the first labelled anomaly and thus allowing time to alert the engineer to an anomaly in advance of the problem. Furthermore it correctly detects one of the other anomalies. In practice, the former is more useful as the engineer can intervene in advance of a machine failure. Precision and recall are 0.004 and 0.25 for R-ESD. Both measures are 0 for SH-ESD. This first anomaly is not explained or discussed in [15]. In fact their analysis does not utilise the first portion of the dataset although it is given and labelled on the `Numenta` GitHub repository. The third anomaly is notoriously difficult to detect as the lead up to this anomaly is a very gradual decline in machine temperature. In terms of CPU time, R-ESD requires approximately 10 seconds to stream 20,000 windows, that is only 0.0005 seconds per window. This is slower than SH-ESD which takes 0.31 seconds. However this is for non-overlapping windows of size 961 rendering the two

reasonably computationally similar (since $(20000/961) \times 0.31 \approx 6.5$ seconds).
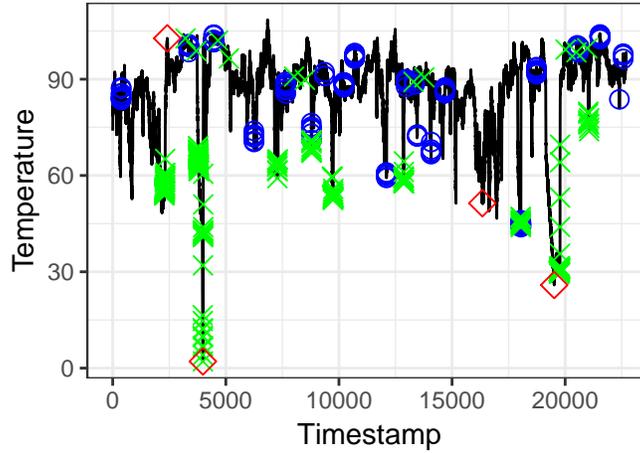

Figure 4. Streaming machine temperature example: Anomalies detected by R-ESD and SH-ESD are shown in green and blue respectively. The number of anomalies tested per window is $k = 10$ for R-ESD (and equivalently max_anoms$= 0.0004$ for SH-ESD). Known anomalies are marked in red.

Finally, the R-ESD algorithm was implemented on the Yahoo EGADS benchmark A3 data (also known as TS-2) [13] with results displayed in Figure 5. This dataset includes outliers and no change points, which is the type of dataset most suitable for good performance of both SH-ESD and R-ESD. As Latpev et al., 2015 admits, there is no best anomaly detection model for all use cases and certain algorithms are best at detecting different types of anomalies. By comparing Figure 5 of [14] and the R-ESD results displayed Figure 5 of this article, it is clear that all methods perform badly in comparison to R-ESD, with F1-scores of less than 0.7 compared with greater than 0.75 for R-ESD.

R-ESD can also be compared with DeepADVote by examining Figure 5 below and Figure 3 of [1]. R-ESD outperforms DeepADVote as demonstrated by a comparable but much less variable F1-score (and recall) for dataset A3, with 25th percentiles as low as 0.3(0.2) for DeepADVote but 0.65(0.5) for R-ESD. R-ESD is by its nature extendable to fit any model or collection of models at every nth streamed datapoint with a trade-off between computational time and streaming capacity.
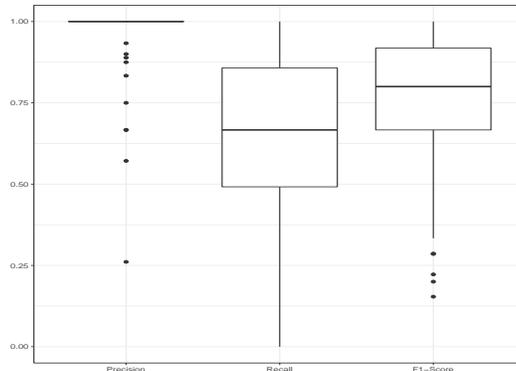

Figure 5. Boxplots of precision, recall and F1-score for the A3 Yahoo EGADS benchmark dataset.

10

5. Conclusion

This paper presents a novel approach to anomaly detection for streaming time series data, which typically exhibits trend and or seasonality. The primary novelty of the R-ESD algorithm is an algebraic manipulation that enables a real-time, low memory, statistically principled version of a widely used approach. It enables the use of multiple recursive updates within the ESD test across rolling windows of data. The major advantage is that this renders the approach feasible for streaming data. In the examples presented, computation times were as little as 0.02 and 0.0005 seconds per window i.e. per streamed datapoint. If required, computation times could be reduced further by implementing a priority queue [12] to reduce memory requirements.

An extension to this algorithm would be to combine more modelling and forecasting approaches for example those give in both DeepAnT [18] and DeepAD [1], where one or multiple models are fit at every newly streamed datapoint. However, there would be a considerable trade-off between computational time and forecasting accuracy which may render it unsuitable for many streaming applications. One option could be to refit the model every n^{th} datapoint or every time an anomaly is detected. Further studies are required to extend the comparison of R-ESD using the Numenta benchmark tests, which explicitly reward early detection. We suspect that R-ESD will perform well by this measure given the results presented for the machine temperature example.

In summary R-ESD is a fast, statistically principled and novel recursive approach to anomaly detection in time series data. It is highly feasible for streaming in real-time. It correctly studentises observations according to the theoretical distribution of the ESD test statistic and it outperforms the `AnomalyDetection` package, thus improving on Twitter's approach. Beyond SH-ESD, R-ESD also outperforms EGADS [14], which uses a combination of many models and methods to detect as many different types of anomalies as possible and DeepAdVote, the most comparable algorithm to ours of the deep-learning based approaches to anomaly detection. While this article focuses on the anomaly detection testing routine,

References

[1] T. S. Buda, B. Caglayan, and H. Assem. Deepad: A generic framework based on deep learning for time series anomaly detection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 577–588. Springer, 2018.

[2] K. Chan and B. Ripley. *TSA: Time Series Analysis. R package version 1.01*, 2012. URL `http://CRAN.R-project.org/package=TSA`.

[3] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[4] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning. STL: A seasonal-trend decomposition. *Journal of official statistics*, 6(1):3–73, 1990.

[5] F. E. Grubbs. Sample criteria for testing outlying observations. *The Annals of Mathematical Statistics*, 21(1):27–58, 1950.

[6] M. Gupta, J. Gao, C. C. Aggarwal, and J. Han. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 26(9): 2250–2267, 2014.

[7] J. Hochenbaum, O. S. Vallis, and A. Kejariwal. Automatic anomaly detection in the cloud via statistical learning. *arXiv preprint arXiv:1704.07706*, 2017.

[8] R. Hyndman and G. Athanasopoulos. *Forecasting: Principles and Practice.*

OTexts, Australia, 2nd edition, 2018.

[9] R. J. Hyndman and Y. Khandakar. Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*, 26(3):1–22, 2008. URL `http://www.jstatsoft.org/article/view/v027i03`.

[10] S. Kandanaarachchi, M. A. Muñoz, R. J. Hyndman, and K. Smith-Miles. On normalization and algorithm selection for unsupervised outlier detection. *Data Mining and Knowledge Discovery*, pages 1–46, 2019.

[11] A. Kejariwal. Introducing practical and robust anomaly detection in a time series, 2015, retrieved from: https://blog.twitter.com/2015/introducing-practical-and-robust-anomaly-detection-in-a-time-series., 2015.

[12] D. E. Knuth. The art of computer programming: Sorting and searching, vol. 3 2nd ed addison wesley longman publishing co. *Redwood City, CA*, pages 458–478, 1998.

[13] N. Laptev and S. Amizadeh. *Yahoo anomaly detection dataset S5*, 2015. URL `http://webscope.sandbox.yahoo.com`. Retrieved on 15 December 2019.

[14] N. Laptev, S. Amizadeh, and I. Flint. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1939–1947, 2015.

[15] A. Lavin and S. Ahmad. Evaluating real-time anomaly detection algorithms - the Numenta anomaly benchmark. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 38–44. IEEE, 2015.

[16] C. Leigh, O. Alsibai, R. J. Hyndman, S. Kandanaarachchi, O. C. King, J. M. McGree, C. Neelamraju, J. Strauss, P. D. Talagala, R. D. Turner, K. Mengersen, and E. Peterson. A framework for automated anomaly detection in high frequency water-quality data from in situ sensors. *Science of The Total Environment*, 664: 885–898, 2019.

[17] K. Mulrennan, J. Donovan, L. Creedon, I. Rogers, J. G. Lyons, and M. McAfee. A soft sensor for prediction of mechanical properties of extruded PLA sheet using an instrumented slit die and machine learning algorithms. *Polymer Testing*, 69: 462–469, 2018.

[18] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed. DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, 7: 1991–2005, 2018.

[19] B. Rosner. On the detection of many outliers. *Technometrics*, 17(2):221–227, 1975.

[20] B. Rosner. Percentage points for a generalized esd many-outlier procedure. *Technometrics*, 25(2):165–172, 1983.

[21] Twitter. *AnomalyDetection. GitHub repository:*, 2015. URL `https://GitHub.com/twitter/AnomalyDetection`. Accessed April 17, 2019.

[22] O. Vallis, J. Hochenbaum, and A. Kejariwal. A novel technique for long-term anomaly detection in the cloud. In *6th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 14)*, 2014.