Online Adaptable Time Series Anomaly Detection with Discrete Wavelet Transforms and Multivariate Gaussian Distributions

Markus Thill, Wolfgang Konen and Thomas Bäck

Abstract In this paper we present an unsupervised time series anomaly detection algorithm, which is based on the discrete wavelet transform (DWT) operating fully online. Given streaming data or time series, the algorithm iteratively computes the (causal and decimating) discrete wavelet transform. For individual frequency scales of the current DWT, the algorithm estimates the parameters of a multivariate Gaussian distribution. These parameters are adapted in an online fashion. Based on the multivariate Gaussian distributions, unusual patterns can then be detected across frequency scales, which in certain constellations indicate anomalous behavior. The algorithm is tested on a diverse set of 425 time series. A comparison to several other state-of-the-art online anomaly detectors shows that our algorithm can mostly produce results similar to the best algorithm on each dataset. It produces the highest average F1-score with one standard parameter setting. That is, it works more stable on high- and low-frequency-anomalies than all other algorithms. We believe that the wavelet transform is an important ingredient to achieve this.

Markus Thill and Wolfgang Konen
TH Köln – University of Applied Sciences, Steinmüllerallee 1, 51643 Gummersbach, Germany
✉ markus.thill@th-koeln.de, wolfgang.konen@th-koeln.de

Thomas Bäck
Leiden University, LIACS, Rapenburg 70, 2311 EZ Leiden, The Netherlands
✉ t.h.w.baeck@liacs.leidenuniv.nl

ARCHIVES OF DATA SCIENCE, SERIES A
(ONLINE FIRST)
KIT SCIENTIFIC PUBLISHING
Vol. -, No. -, -

DOI 10.5445/KSP/XXXXXXXX/XX
ISSN 2363-9881

(cc) BY-SA

1 Introduction

Up till today, anomaly detection in general and especially for time series remains a challenging task. A successful anomaly detector should fulfill the following requirements to be useful in practice: (i) detect anomalies in an unsupervised manner, (ii) operate online and adaptively, and (iii) work robustly on quite different time series data.

Requirement (i) arises from the fact that it is usually not possible to collect enough anomalous data in a training phase and that it is cumbersome in practice to even separate in training and operational phase. Instead it is desirable to have an algorithm observing and learning from the ’normal’ data stream and detecting significant deviations as anomalies. Requirement (ii) comes from the fact that time series data in practice need not to be stationary and/or can be too big for batch processing. The most notable advantage of online algorithms might be their adaptive capabilities, which allow them to learn in non-stationary environments and to adapt to concept drifts or concept changes. Most state-of-the-art anomaly detectors (see Sec. 1.1) will usually fulfill (i) and (ii). Requirement (iii) is less obvious, nevertheless of great practical relevance: It is desirable to have one algorithm for diverse data: sometimes the data are high-frequent (spiky, e. g. network traffic data), sometimes the data are medium- or low-frequent (e. g. sensor signals). In our recent work (Thill et al, 2017) it was found to our surprise that most state-of-the-art algorithms are either good in one domain or the other. This stirred the work presented in this paper which uses wavelet transforms to generate features in diverse frequency ranges. The underlying research question is: Is it possible to propose *one* online anomaly detection algorithm which works robustly on a *diverse* set of benchmarks?

In the following sections we extend our recent work (Thill et al, 2017) and introduce an unsupervised anomaly detection algorithm based on the discrete wavelet transform (DWT) which operates fully online and shows robust performance on several benchmarks, using only one parameter setting.

1.1 Related Work

Although many anomaly detection techniques have been developed over the past years, as for example surveyed in (Chandola et al, 2009) and (Patcha and Park, 2007), not many approaches utilize wavelet transforms for detecting anomalies in time series signals. From those techniques found in the literature, most are designed for high-frequency anomaly detection (e.g. in network traffic data), such as (Kim et al, 2004; Kwon et al, 2006) and (Lu and Ghorbani, 2009). The early work of Alarcon-Aquino and Barria (2001); Alarcon Aquino (2003) describes anomaly detection based on non-decimating wavelet transforms. Kanarachos et al. (Kanarachos et al, 2015) developed an anomaly detection algorithm for time series, based on wavelets, neural networks and Hilbert transforms. The algorithm was tested on a relatively simple benchmark, including two synthetic time series.

In this work we will compare the results of our proposed online anomaly detection method to the state-of-the-art algorithms NuPic (George and Hawkins, 2009) and ADVec (Vallis et al, 2014), which both are open-source available. As benchmark data we use the Numenta Anomaly Benchmark (Lavin and S. Ahmad, 2015) and Yahoo's Webscope S5 benchmark (Laptev and Amizadeh, 2015).

2 Methods

In this section we describe an online version of an algorithm based on **D**iscrete **W**avelet **T**ransforms with **M**aximum **L**ikelihood **E**stimation for **A**nomaly **D**etection in time series, in short DWT-MLEAD.

2.1 Discrete Wavelet Transforms

Wavelet transforms (Meyer and Salinger, 1995) are used to construct a frequency representation for a signal by finding a representation of the signal in terms of a wavelet function (a so called mother wavelet, e.g. a Haar wavelet), which is scaled (stretched and shrinked) in order to capture different frequency information and shifted along the time axis. Wavelet transforms allow to retrieve a time series signal representation which is accurate in both the time and frequency domain. In this sense wavelet transforms are an interesting alternative to classical approaches such as (short-time) Fourier transforms, where one can either achieve a high resolution in the time domain or frequency domain, but not in both at the same time. For sampled time series data, often the so called discrete wavelet transform (DWT) is applied, which has linear time complexity. Usually a decimating DWT is performed, in which the filtered series are downsampled. The DWT decomposes the original time series into so called approximation and detail coefficients which are arranged in different levels. Due to the decimating (downsampling) property of the DWT one can represent both coefficient sets in two binary tree structures. In this work we apply a decimating DWT to the time series using Haar wavelets. Other wavelets are also applicable, but require some additional considerations. Since lower levels of the DWT usually do not contain patterns which are useful for anomaly detection, only the L highest levels (L is a parameter of the algorithm) are considered, where $\ell = L-1$ describes to lowest considered level and $\ell = 0$ addresses the highest possible level, which is the original time series and which only contains the detail coefficients. The DWT-MLEAD algorithm utilizes both the detail coefficients $d_{n,\ell}$ and approximation coefficients $c_{n,\ell}$.

For the online implementation of the algorithm, a strictly causal computation scheme is adhered to: For example, two data points in the original time series have to be collected first before the next coefficient in level $\ell = 1$ can be computed. Similarly, 2^ℓ data points from the original time series are necessary to compute the next coefficient in level ℓ.

2.2 Sliding Windows

Sliding windows are often used in practice to model local temporal relationships within time series. Our algorithm employs a sliding window for each level of the DWT tree. The length w_ℓ of the window is level-dependent and is computed as $w_\ell = \max\{1, \lfloor b^{o-\ell} \rfloor\}$ where $b, o \in \mathbb{R}$ are two parameters of the algorithm. As soon as a new coefficient in level ℓ is available ($c_{n,\ell}$ or $d_{n,\ell}$), the corresponding window is slid one further and the new window embedding is collected and passed to a model, which estimates the likelihood of observing such a vector (as described in the following sections). Unlikely vectors would indicate unusual behavior on the corresponding DWT level. As already mentioned before, the sliding windows at lower levels are moved with a slower rate than those on higher levels, since new coefficients are only generated after every 2^ℓ time steps in the original time series. Anomaly detection starts after an initial transient phase, when the sliding windows can be completely filled.

2.3 Online Estimation of Gaussian Distributions

In order to distinguish between normal and unusual patterns in the individual levels of the DWT, our algorithm estimates a multivariate Gaussian distribution for each considered level. This is done separately for the approximation and detail coefficients ($c_{n,\ell}$ and $d_{n,\ell}$). The dimension of the Gaussian distribution depends on the length of the sliding window used in each level of the DWT. Each Gaussian distribution is parameterized by a mean vector $\hat{\boldsymbol{\mu}} \in \mathbb{R}^{w_\ell}$ and a covariance matrix $\hat{\boldsymbol{\Sigma}} \in \mathbb{R}^{w_\ell \times w_\ell}$ which can be found by using maximum likelihood estimation (MLE) (Thill et al, 2017). Since the DWT-MLEAD algorithm operates in an online fashion, the parameter estimations also have to be updated incrementally for each new data point. For this purpose we use an exponentially decaying weighted estimator with an forgetting factor $\lambda \in (0,1]$. The forgetting factor controls at which rate past observations fade out over time. A value of λ close to 1 results in an algorithm with a very long memory, whereas small values (usually not smaller than 0.9) can significantly limit the memory of the estimator. By allowing the estimator to gradually forget historic information, the algorithm can adapt to new concepts in the data stream. Furthermore, with $\lambda < 1$ we can prevent (under most conditions) a numeric overflow of the required accumulator (the sum of squares of differences from the current mean). However, forgetting can also lead to a higher variance in the parameter estimates. The pseudo-code of the estimator can be found in Alg. 2, lines 1 – 8. Note that it is not actually necessary to compute the covariance matrix, since only its matrix inverse is required in later steps. Therefore, we directly estimate the inverse of the sum of squares of differences from the current mean $\boldsymbol{M}_n^{-1}$. Since the inverse $\boldsymbol{M}_n^{-1}$ has to be re-computed for every new data point, which can be computationally expensive for larger dimensions, we use the Sherman-Morrison formula (Sherman and Morrison, 1950) – a special case of the Woodbury matrix identity (Woodbury, 1950) – to incrementally update $\boldsymbol{M}_n^{-1}$. The inverse of the covariance matrix is given by $\hat{\boldsymbol{\Sigma}}_n^{-1} = W_n \boldsymbol{M}_n^{-1}$.

2.4 Detecting Events in the DWT Tree and Anomaly Detection

Since DWT-MLEAD estimates a multivariate Gaussian distribution for every set of DWT-coefficients on the levels $\ell \in [0,1,\ldots,L]$, it is possible to examine each newly observed value $c_{n,\ell}$ and $d_{n,\ell}$ in the context of its current sliding

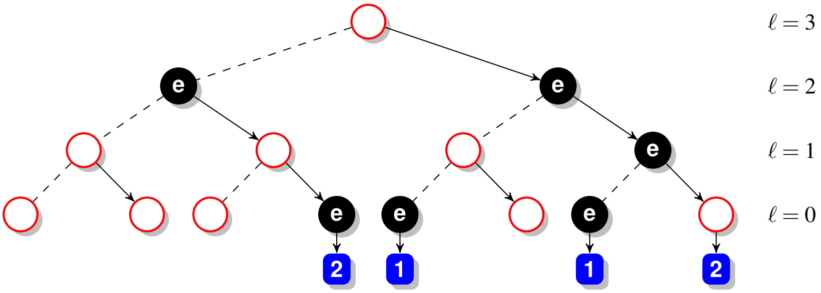

Fig. 1 Detecting anomalies with leaf counters. All coefficients (except on the leafs) are always computed bottom-up, based on two child nodes (connected with one dashed and one solid edge). Along the vertical axis are the DWT levels ℓ, along the horizontal axis are the time indices n of the coefficients of the DWT. E. g., the leftmost event e comes from either an unusual $c_{n,2}$ or $d_{n,2}$. Each event is passed down the tree only along the solid edges (causal computation) and increases the right-most leaf counter (blue rectangle) connected with the e node.

window, in order to detect unusual patterns. For each new data point the current window embed vector is determined and the squared Mahalanobis distance $m_{\mathbf{x}_n}$ to the center of the Gaussian is computed for this vector. Subsequently, this distance is compared to a threshold m_ε. Since a Gaussian random variable has a squared Mahalanobis distance to its mean, which is Chi-squared (χ^2) distributed with w_ℓ degrees of freedom, we can determine m_ε by simply computing the $(1-\varepsilon)$-quantile of the χ^2-distribution (function PREDICT in Algorithm 2, lines 10–15). If the Mahalanobis distance $m_{\mathbf{x}_n}$ exceeds the threshold m_ε, the current instance $c_{n,\ell}$ or $d_{n,\ell}$ is flagged as unusual and an event e is passed down the DWT tree, as illustrated in Fig. 1. Events arriving at the leaf nodes are summed up in a global, exponentially decaying event counter E_i (Algorithm 1, line 25). If the activity in a subtree of the DWT exceeds a certain limit, hence, if many events are produced in a short time, E_i will increase fast. As soon as E_i is larger than a specified threshold B, an anomaly will be fired and the instance i in the time series will be flagged. In order to avoid many detections in a short time, a new anomaly cannot be fired again until E_i has faded away and falls below a threshold.

In order to capture extreme outliers, a simple heuristic is used after $i > 500$, which tracks the maximum and minimum value observed in the time series and flags a point as anomalous, if it exceeds the current maximum/minimum by more than 20% of the min-max range.

Algorithm 1 An online version of DWT-MLEAD, an anomaly detection algorithm using the Discrete Wavelet Transform.

```
Define parameters:
        $L$: maximum number of levels considered in the DWT
        $b$, $o$: for the computation of the sliding window sizes $w_\ell$
        $\lambda$: forgetting factor for the estimation of the Gaussian distributions
        $\varepsilon$: quantile of $\chi^2$-distribution
        $B$: threshold for global event counter that triggers an anomaly

Initialize:
        Set window sizes for each level: $w_\ell = \max\{1, \lfloor b^{o-\ell} \rfloor\}$
        Global event counter: $E_0 = 0$
        Discount factor: $\gamma = \frac{w_L - 1}{w_L + 1}$
        Allow to trigger anomaly with: $A = \text{true}$
        Initialize all $P_0^{(c,\ell)}$ and $P_0^{(d,\ell)}$ with the tuple $(W_0, \hat{\boldsymbol{\mu}}_0, \boldsymbol{M}_0^{-1}, \boldsymbol{M}_0)$, where:
                $W_0 \in \mathbb{R}$, $\hat{\boldsymbol{\mu}}_0 \in \mathbb{R}^{w_\ell}$ and, $\boldsymbol{M}_0^{-1}, \boldsymbol{M}_0 \in \mathbb{R}^{w_\ell \times w_\ell}$
                $W_0 = 0$, $\hat{\boldsymbol{\mu}}_0 = \mathbf{0}$, $\boldsymbol{M}_0^{-1} = \boldsymbol{M}_0 = \mathbf{I}$

function DWTMLEAD($i$, $y_i$)                    ▷ where $y = (y_1, y_2, \ldots)$ is a streaming time series
    Determine $\ell' = \min(L-1, \max\{\ell^* \in \mathbb{N}_0 \mid i \mod 2^{\ell^*} = 0\})$
    for all $\ell \in \{0, \ldots, \ell'\}$ do
        $n = i/2^\ell$
        Compute DWT coefficients $c_{n,\ell}$ and $d_{n,\ell}$                    ▷ if not already present
        $\boldsymbol{x}_n^{(c)} = \left(c_{n-w_\ell+1,\ell} \cdots c_{n,\ell}\right)^\intercal$ , $\boldsymbol{x}_n^{(d)} = \left(d_{n-w_\ell+1,\ell} \cdots d_{n,\ell}\right)^\intercal$                    ▷ sliding windows
        $P_n^{(c,\ell)} = \text{UPDATE}(P_{n-1}^{(c,\ell)}, \boldsymbol{x}_n^{(c)}, \lambda)$ , $P_n^{(d,\ell)} = \text{UPDATE}(P_{n-1}^{(d,\ell)}, \boldsymbol{x}_n^{(d)}, \lambda)$
        $e_\ell = \text{PREDICT}(P_{n+1}^{(c,\ell)}, \boldsymbol{x}_n^{(c)}, \varepsilon) + \text{PREDICT}(P_{n+1}^{(d,\ell)}, \boldsymbol{x}_n^{(d)}, \varepsilon)$
    $E_i = \gamma E_{i-1} + \sum_{j=0}^{\ell'} e_j$                    ▷ Adjust global event counter
    $a_i = \begin{cases} \text{true, } \textbf{if } A \wedge E_i \geq B \\ \text{false, } \textbf{otherwise} \end{cases}$                    ▷ Flag anomaly at time step $i$, if threshold is exceeded
    if $a_i$ then $A = \text{false}$
    if $E_i < \frac{2}{3}B$ then
        $A = \text{true}$                    ▷ Allow new anomaly, if event-counter value falls below threshold
    return $a_i$
```

3 Experimental Setup

3.1 The Benchmarks

In order to evaluate the performance of the DWT-MLEAD algorithm and compare the results to other algorithms, we use a very diverse benchmark consisting of in total 425 time series. The benchmark is composed of the Yahoo Webscope S5 data (Laptev and Amizadeh, 2015) and the Numenta Anomaly Benchmark

Algorithm 2 Helper functions for Algorithm 1.

```
function UPDATE($P_{n-1}$,$\boldsymbol{x}_n$,$\lambda$)    ▷ $\boldsymbol{x}_n \in \mathbb{R}^{w_\ell}$, where $w_\ell$ is the size of the window at scale $\ell$
    $(W_{n-1}, \hat{\boldsymbol{\mu}}_{n-1}, \boldsymbol{M}_{n-1}^{-1}, \boldsymbol{M}_{n-1}) = P_{n-1}$    ▷ Matrix $\boldsymbol{M}_{n-1}$ is optional (debugging purposes)
    $W_n = \lambda W_{n-1} + 1$
    $\boldsymbol{\Delta}_n = \boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_{n-1}$
    $\hat{\boldsymbol{\mu}}_n = \hat{\boldsymbol{\mu}}_{n-1} + \frac{1}{W_n}\boldsymbol{\Delta}_n$
    $\boldsymbol{M}_n = \lambda \boldsymbol{M}_{n-1} + \boldsymbol{\Delta}_n(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_n)^\top$    ▷ Optional, since only inverse $\boldsymbol{M}_n^{-1}$ is required later
    $\boldsymbol{M}_n^{-1} = \frac{1}{\lambda}\boldsymbol{M}_{n-1}^{-1} - \frac{\frac{1}{\lambda}\boldsymbol{M}_{n-1}^{-1}\boldsymbol{\Delta}_n(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_n)^\top \boldsymbol{M}_{n-1}^{-1}}{\lambda + (\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_n)^\top \boldsymbol{M}_{n-1}^{-1}\boldsymbol{\Delta}_n}$    ▷ Inverse using the Sherman-Morrison Formula
    return $(W_n, \hat{\boldsymbol{\mu}}_n, \boldsymbol{M}_n^{-1}, \boldsymbol{M}_n)$    ▷ Return updated parameters

function PREDICT($P_n$,$\boldsymbol{x}_n$,$\varepsilon$)    ▷ $\boldsymbol{x}_n \in \mathbb{R}^{w_\ell}$, where $w_\ell$ is the size of the window at scale $\ell$
    $(W_n, \hat{\boldsymbol{\mu}}_n, \boldsymbol{M}_n^{-1}, \boldsymbol{M}_n) = P_n$
    $m_{\boldsymbol{x}_n} = W_n(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_n)^\top \boldsymbol{M}_n^{-1}(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_n)$    ▷ Mahalanobis distance of $\boldsymbol{x}_n$ to $\hat{\boldsymbol{\mu}}_n$
    $m_\varepsilon = \chi^2_{1-\varepsilon}(w_\ell)$    ▷ Threshold: upper $\varepsilon$-quantile of $\chi^2$-distribution
    $e_n = \begin{cases} 1, & \textbf{if } m_{\boldsymbol{x}_n} > m_\varepsilon \\ 0, & \textbf{otherwise} \end{cases}$    ▷ Binary event flag
    return $e_n$    ▷ Unusual data points will cause an event in the DWT-tree
```

(NAB) (Lavin and S. Ahmad, 2015), which are both publicly available. The Webscope S5 benchmark (with overall 572,966 data points) is split again into the 4 datasets A1, A2, A3 and A4 containing 67, 100, 100 and 100 time series. While the A1 data consists of real data, mostly from computational services, A2 to A4 contain synthetic time series with increasing complexity. On average, each time series has approximately 1,500 instances.

The NAB data contains 58 time series (with in total 365,558 data points), with the majority (47 time series) coming from real world applications such as server monitoring, network utilization, sensor readings from industry and social media statistics. The longest time series contains 22,695 and the series contain approximately 6,300 instances on average. The ground truth anomaly labels are available for all considered time series, however, it is important to note that they are not passed to the anomaly detection algorithms at any time and only used to assess the algorithm's performance afterwards. Examples for each dataset are shown in Fig. 2.

3.2 Algorithmic Setup

In this work, we compare DWT-MLEAD to two other online anomaly detection algorithms. For each algorithm *one* standard parameter setting is chosen which

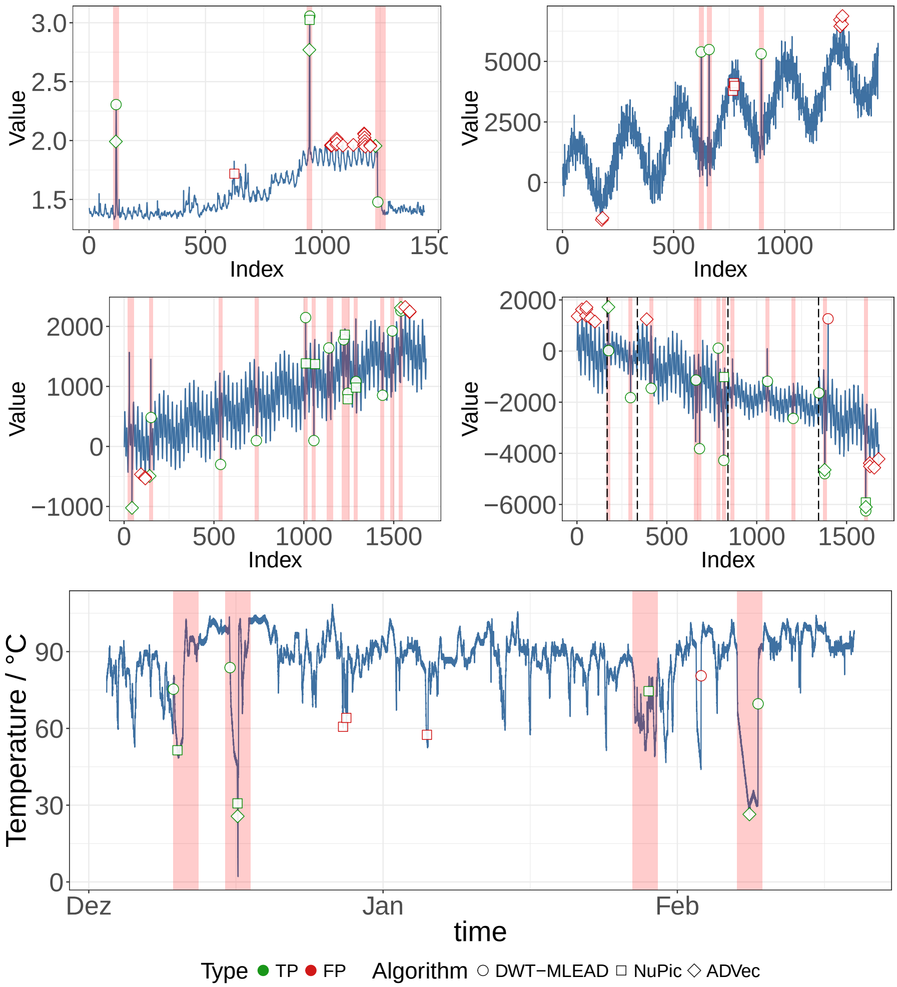

Fig. 2 Example time series taken from the Yahoo Webscope S5 data and the Numenta Anomaly Benchmark (NAB). In each graph the real anomalies are indicated by the light-red shaded areas. Three algorithms are tested on this data and the individual detections are shown with different symbols. The color of the symbol indicates if the detections were correct (green) or false (red).
Top two rows: One example each from the A1–A4 data. The dashed vertical lines in the A4 data indicate concept changes, which should also be detected by the anomaly detectors.
Bottom: Example time series taken from the NAB data. The graph shows the temperature sensor data of an internal component of a large industrial machine over its last months before a catastrophic failure occurs end of February. The second anomaly (mid of December) is a planned shutdown of the machine.

is then used for all experiments across all datasets. Only an anomaly threshold parameter is varied for each algorithm and dataset, in order to balance precision and recall in a way that the F_1-score is maximized.

DWT-MLEAD As described in Sec. 2, in total 6 parameters have to be selected by the user. In order to find an appropriate setting, we did not systematically tune the parameters. Instead, we generated 60 design points using latin hypercube sampling (LHS) and evaluated the algorithm for these points. The setting $B = 2.20$, $b = 2.27$, $o = 6$, $L = 5$, $\lambda = 0.972$ achieved the highest average F_1-score and will be used throughout the rest of this paper. The parameter ε is used as anomaly threshold and is adjusted in the range $\varepsilon \in [10^{-6}, 10^{-1}]$.

NuPic Numenta's online anomaly detection algorithm (George and Hawkins, 2009) has a large set of parameters. The parameters can be tuned using an inbuilt swarming (Ahmad, 2017) algorithm. However, we found that swarming does not improve the results significantly compared to a standard configuration, as used in (Lavin and S. Ahmad, 2015). Similarly to DWT-MLEAD, an anomaly threshold can be varied in the interval $[0, 1]$ to control the sensitivity of the algorithm.

ADVec This algorithm was developed by Twitter (Vallis et al, 2014) and is based on the generalized ESD test, combined with robust statistical approaches and piecewise approximation. Mainly, three parameters are required, which we tuned to achieve the highest average F_1-score. The first parameter is the period-length, which is set to the value 40. The second parameter, $\max_{anoms} = 0.003$, specifies the maximum number of anomalies that the algorithm will detect as a percentage of the data. The last parameter α describes the level of statistical significance with which to accept or reject anomalies. We use this parameter as anomaly threshold for ADVec and adjust it in the range $\alpha \in [10^{-6}, 320]$ for our experiments.

3.3 Algorithm Evaluation

In order to compare the performance of the different algorithms on the described benchmarks, suitable performance metrics are required. Similarly to binary classification tasks, every instance in the time series can be classified either as normal or as anomalous. A correctly identified anomaly will be counted as a true positive (TP), whereas a point incorrectly flagged as anomalous will be considered as a false positive (FP) and a missed anomaly as a false negative (FN). The number of data points in a time series which is correctly predicted as normal (true negatives or TN) is usually not meaningful and will therefore not be used for evaluation purposes. Furthermore, since most anomalies in time series are not point-anomalies but span over longer time-intervals, a time frame of appropriate length, the so called anomaly window, is used to describe each anomaly. Consequently, several detections within an anomaly window will only be counted as one TP and a missed anomaly window will only be counted as one FN. From the aforementioned quantities, the well known metrics precision, recall and F_1-score are derived, whereby the latter is the harmonic mean of precision and recall. The average metrics in column **Avg** of Table 1 are the metrics' mean over the five datasets A1–A4 and NAB.

4 Results

The main results of our experiments are summarized in Tab. 1. DWT-MLEAD achieves on all datasets the highest F_1-score. NuPic has a slightly better precision on A1, but on A2, A3 and A4 the difference in all three metrics is large in favor of DWT-MLEAD. One reason, among others, for the weak performance of NuPic and ADVec could be that the time series in both datasets contain many anomalies, occurring in part at the very beginning of each time series. Hence, the algorithms have to be up-and-running much faster and have to be able to detect anomalies in short time intervals. Furthermore, the A4 time series contain many concept changes, where amplitudes, seasonalities and noise abruptly change. In order to handle such concept changes, a strong online adaptability is required. For the NAB data, the difference in F_1-score between NuPic and DWT-MLEAD is not that apparent, although there is a slight advantage for our algorithm. Overall, we can observe in column **Avg** that DWT-MLEAD achieves the highest average values for all three metrics.

Since Tab. 1 only captures the results for one specific setting of the algorithms anomaly thresholds, we also measured precision and recall for a wide range of thresholds and plotted them against each other, as shown in Fig. 3. The overall picture mostly corresponds to the results shown in Tab. 1. Only for the NAB data we can observe, that for recall values in the range $[0.5, 0.75]$ NuPic achieves a higher precision and outperforms DWT-MLEAD.

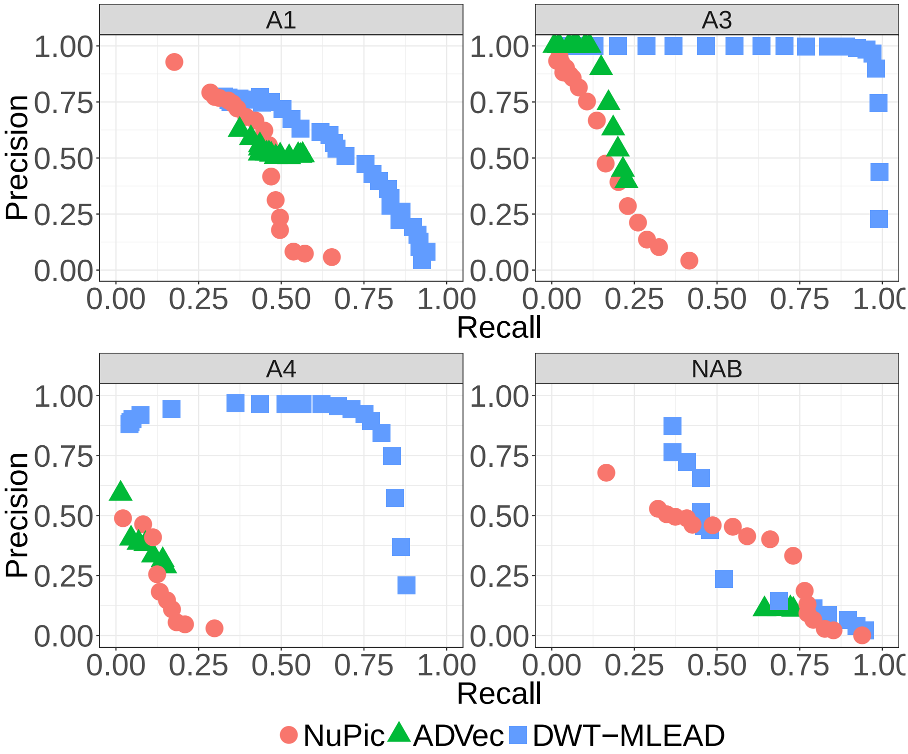

Fig. 3 Multiobjective plot for Yahoo's Webscope S5 benchmark and the Numenta Anomaly benchmark. The graph for the A2 data is not shown here since the results are very similar to the A3 data.

Table 1 Results for various algorithms on the datasets A1–A4 and NAB. Shown are the metrics precision (how many percent of the detected events are true anomalies), recall (how many percent of the true anomalies are detected) and F_1. All algorithms have their threshold for each dataset chosen such that F_1 is maximized. Each algorithm uses otherwise one standard parameter set on all data sets.

	Precision, Recall **F_1-Score**					
Algorithm	**A1**	**A2**	**A3**	**A4**	**NAB**	**Avg**
DWT-MLEAD	0.6, 0.65 **0.62**	1, 0.98 **0.99**	0.96, 0.97 **0.97**	0.92, 0.75 **0.83**	0.66, 0.45 **0.54**	0.8, 0.76 **0.79**
NuPic	0.62, 0.45 0.52	0.59, 0.42 0.49	0.39, 0.2 0.27	0.41, 0.11 0.18	0.4, 0.66 0.5	0.32, 0.37 0.39
ADVec	0.51, 0.56 0.54	0.66, 0.6 0.63	0.54, 0.2 0.29	0.29, 0.15 0.2	0.11, 0.72 0.2	0.32, 0.45 0.37

5 Discussion

Although algorithm DWT-MLEAD could produce good results on the investigated benchmarks, it still has several limitations which leave room for improvement: (1) For our experiments we only used the relatively simple Haar wavelet. In practice, it could be helpful to use more complex wavelets. (2) Due to the strictly causal design of the algorithm, events occurring in the DWT-tree might be asymmetrically distributed along the leaf counters (Fig. 1). More events will tend to arrive at the leaf nodes on the right side of each sub-tree, which might lead to undesired effects.[1] (3) One might object that the Gaussian distribution may not be the best choice to model the data. Other (perhaps multimodal) distributions might be more effective. To test this, we made some runs with Gaussian mixture models (GMM) which are capable to model more complex distributions. So far, however, these runs resulted in only marginal improvements. This supports that Gaussian distributions are well usable in our case.

It is worth mentioning that DWT-MLEAD proved to perform robustly on all time series, without ever showing numerical instabilities from the matrix updates (function UPDATE in Algorithm 2).

6 Conclusion & Future Work

In this paper we introduced the relatively simple but effective DWT-MLEAD algorithm for online anomaly detection in time series. We found that especially the discrete wavelet transform (DWT) can be an important tool to generate meaningful features across many different frequency scales. Empirical results on a large dataset with 425 time series containing both long-term and short-term anomalies, show that DWT-MLEAD is more robust than other state-of-the-art anomaly detectors: Using only *one* fixed parameter setting, DWT-MLEAD achieved an average F_1 twice as large as for the other two algorithms (Table 1). Furthermore, the online adaptability of the DWT-MLEAD algorithm appears to be beneficial in the presence of concept drifts and/or changes, as the results on the A4 data of Yahoo's Webscope S5 benchmark suggest. Our anomaly

[1] We note in passing that we performed runs with an algorithmic variant where we treated each leaf symmetrical: We wait until an L-subtree is complete, then we collect all events (along the dashed lines in Fig. 1 as well) and process them. The price to pay is a certain delay for some leafs and a deviation from the strict online scheme. The results in terms of precision-recall-metrics are a bit better for NAB and a bit worse for A4. Overall, the difference is only marginal.

detection algorithm does not require labeled training data, it infers from the unlabeled data of each time series what is normal and what is anomalous.

As future work we are planning to improve several aspects of our algorithm: Currently, only simple Haar wavelets are used for the algorithm; experiments with other wavelets might lead to an significantly increased performance. Another interesting direction of work could be – although we could achieve good results with simple multivariate Gaussian distributions – to investigate other unsupervised learning approaches in order to learn more accurate models of the underlying distribution of the time series data. Furthermore, we are planning to further reduce the sensitivity of DWT-MLEAD towards its parameters, for example with automatic parameter tuning methods.

Finally, a look on Table 1 shows that the NAB dataset is a tough benchmark: All tested algorithms are far from being perfect on that dataset, having $F_1 < 0.55$, i. e. there is still room for improvement.

References

Ahmad S (2017) Running swarms. URL `http://nupic.docs.numenta.org/0.6.0/guide-swarming.html`

Alarcon Aquino V (2003) Anomaly detection and prediction in communication networks using wavelet transforms. PhD thesis, Imperial College London

Alarcon-Aquino V, Barria JA (2001) Anomaly detection in communication networks using wavelets. IEE Proceedings-Communications 148(6):355–362, DOI 10.1049/ip-com:20010659

Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: A survey. ACM computing surveys (CSUR) 41(3):15, DOI 10.1145/1541880.1541882

George D, Hawkins J (2009) Towards a mathematical theory of cortical microcircuits. PLoS Comput Biol 5(10):e1000,532, DOI 10.1371/journal.pcbi.1000532

Kanarachos S, Mathew J, Chroneos A, Fitzpatrick M (2015) Anomaly detection in time series data using a combination of wavelets, neural networks and Hilbert transform. In: International Conference on Information, Intelligence, Systems and Applications (IISA), pp 1–6, DOI 10.1109/IISA.2015.7388055

Kim SS, Reddy AN, Vannucci M (2004) Detecting traffic anomalies using discrete wavelet transform. In: International Conference on Information Networking, Springer, pp 951–961, DOI 10.1007/978-3-540-25978-7_96

Kwon D, Ko K, Vannucci M, Reddy AN, Kim S (2006) Wavelet methods for the detection of anomalies and their application to network traffic analysis. Quality and Reliability Engineering International 22(8):953–969, DOI 10.1002/qre.781

Laptev N, Amizadeh S (2015) Yahoo anomaly detection dataset S5. URL `http://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70`

Lavin A, S Ahmad S (2015) Evaluating real-time anomaly detection algorithms – the Numenta anomaly benchmark. In: IEEE Conference on Machine Learning and Applications (ICMLA2015), DOI 10.1109/ICMLA.2015.141

Lu W, Ghorbani AA (2009) Network anomaly detection based on wavelet analysis. EURASIP Journal on Advances in Signal Processing 2009:4, DOI 10.1155/2009/837601

Meyer Y, Salinger D (1995) Wavelets and Operators, Cambridge Studies in Advanced Mathematics, vol 1. Cambridge University Press, DOI 10.1017/CBO9780511623820

Patcha A, Park JM (2007) An overview of anomaly detection techniques: Existing solutions and latest technological trends. Comput Networks 51(12):3448–3470, DOI 10.1016/j.comnet.2007.02.001

Sherman J, Morrison WJ (1950) Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. The Annals of Mathematical Statistics 21(1):124–127, DOI 10.1214/aoms/1177729893

Thill M, Konen W, Bäck T (2017) Time series anomaly detection with discrete wavelet transforms and maximum likelihood estimation. In: Valenzuela O, Rojas I, et al (eds) Intern. Conference on Time Series (ITISE)

Vallis O, Hochenbaum J, Kejariwal A (2014) A novel technique for long-term anomaly detection in the cloud. In: 6th USENIX Workshop on Hot Topics in Cloud Computing, Philadelphia, PA

Woodbury MA (1950) Inverting modified matrices. Memorandum Report 42, Statistical Research Group, Princeton University, Princeton, NJ,, DOI 10.1137/1031049