Anomaly Detection in Univariate Time Series: An Empirical Comparison of Machine Learning Algorithms

Sina Däubener[1], Sebastian Schmitt[2], Hao Wang[3][0000−0002−4933−5181], Peter Krause[1], and Thomas Bäck[3][0000−0001−6768−1478]

[1] divis intelligent solutions GmbH, Joseph-von-Fraunhofer-Str. 20, 44227 Dortmund, Germany `daeubener@divis-gmbh.de`

[2] Honda Research Institute Europe GmbH, Carl-Legien-Strasse 30, D-63073 Offenbach/Main, Germany `sebastian.schmitt@honda-ri.de`

[3] LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, Netherlands {`h.wang,t.h.w.baeck`}`@liacs.leidenuniv.nl`

Abstract. This paper presents an empirical comparison of common machine learning and statistical methods applied to univariate time series with the purpose of detecting anomalies. Based on the assumption that anomalies are infrequently observed and non predictable states, we use regression algorithms to identify these points. In particular we applied random forests, support vector machines, k-nearest neighbour regression, artificial neural networks, Gaussian Processes, Twitter's anomaly detection method AdVec and an ARIMA model. Each algorithm is trained on the complete dataset to learn normal behaviour where the default hyperparameters are used. To dismantle the unpredictable states the generalized extreme studentized test is applied on the residuals. We compare this method on publicly available data sets with labeled anomalies covering a total of 419 time series. Even though the training process of the anomaly detection systems is unsupervised, the labels are available in the datasets, so that well-established measures of classification accuracy are applicable for evaluating the performance of the applied algorithms. The results demonstrate that all algorithms show comparable performance with a slight favor for Gaussian processes and support vector machines. The simple method used here delivered an F1 score higher or equal to 0.8 in 36% of the time series data sets.

Keywords: Anomaly detection · univariate time series · machine learning

1 Introduction

Anomaly detection has received increasing attention with the progress of digitization in private life and Industry 4.0. There are multiple application areas ranging from detecting unusual user behaviour in social media or network intrusion detection to industrial applications, e.g. in production process control. These diverse fields offer different definitions of anomalous data. What is common for many of them is that anomalies are defined as **observations which occur infrequently and are significantly different from other observations** [13]. We differentiate between two groups of methods for finding anomalies:

1. Direct approaches like clustering, classification or distance/density based methods: Anomalies are roughly considered to be either far away from cluster centers, to form a very small class/cluster of its own, to be at far distance from their nearest neighbours [12], or to have a high probabilistic distance [18].
2. Indirect or more precisely residual based approaches: The normal behavior is learned and modeled. Based on these models, predictions are made and the deviation between observed and predicted value is used to decide whether an observation is anomalous.

Many direct anomaly detection methods are described in [13]. In addition, there are some newer methods like hybrid isolation forests [21], which is a combination of separating data points while simultaneously using the density of the data distribution. Another method is based on robust principal component analysis [23], which is for example used for network intrusion detection. However, within this paper we focus on residual based methods, where the creation of a prediction model comes before the anomaly detection task. We use standard regression methods and apply them on univariate time series where previously observed points serve as input parameters [22]. The characteristics of a time series is that it is "a sequence of observation consecutive in time", which leads to the formal description of a time series [9] as a vector $\mathbf{y} = (y_{t_1}, y_{t_1}, ..., y_{t_n})$ of n discrete observations at consecutive time points $t_i < t_{i+1}$ $\forall i \in \{1, ..., n\}$. In the following, we will write y_i instead of y_{t_i} for simplicity of the notation. A popular method for time series regression is based on auto regressive models and dates back to the 1960s. These models use previous observed data points and assume a linear relationship to the next instance. Autoregressive (integrated) moving average (AR(I)MA) regression [9] is an extension for additional feature generation. As an alternative, machine learning approaches can be applied to time series by generating features for classification models [4] or using previous (transformed) values as input variables [2,7].
There are a few other algorithms which are based on an residual approach and are explicitly designed for anomaly detection in time series. Prominent examples are Twitter's AdVec algorithm [17] and Numenta's anomaly detection method based on hierarchical temporal memory [1]. We found that in industry the first obstacle in anomaly detection is to label anomalous instances in existing data. Therefore, the approach presented here is relevant for real world applications by highlighting possible anomalies without extensive data pre-processing.
The remainder of this paper is structured as follows: The modeling algorithms are introduced in section 2. Section 3 describes the data sets used for testing and comparing algorithms. In section 4, the experimental results are presented and discussed, and section 5 provides our conclusions and an outlook for future research.

2 Modelling Algorithms

In this paper, we follow the assumption that anomalies are unpredictable states and therefore punctual and collective anomalies can be detected with a regression model. When learning a regression model based on a fixed small window w of consecutive previous points with the goal to predict the next instance, each regression method learns indirectly the small pattern of this window. A point anomaly can be detected if the predicted value is significantly different from the observed one. It is possible to detect collective anomalies if the small input pattern has not been observed multiple times and therefore the prediction is expected to deviate from the real observation. Because of the limitation of the window size, deviations in pattern length which exceed the input window can not be detected. Without analyzing and adapting to each time series this is a compromise to make. In the simplistic approach used here we set $w = 10$, with which the computations are still feasible. Given the time series $y = (y_1, ..., y_n)$ with window $w = 10$, we use $\{y_i, ..., y_{i+w-1}\}$ as input to predict y_{i+w} for all $i \in \{1, ..., n - w\}$. This way we use all the data to fit a regression model in the first step. Next, this model is used to predict the data it was fitted on. Each method thereby fits the training data to its best of capabilities regarding its default hyperparameter settings. Even though this is regarded as learning by heart associated with overfitting, and normally cross-validation, separate test- and training data set or leave-one-out approaches are common to prevent this overfitting, we utilize this to detecting anomalies. We want the data to be learned as much as possible by each algorithm and then thes the ability to detect unsupervised anomalies without any pre-processing of the data set. After the model is fit we obtain the predictions $\hat{y}_{w+1}, \hat{y}_{w+2}, ..., \hat{y}_n$. For flagging possible anomalies we use the residuals $r_i = y_i - \hat{y}_i$ (difference between observed value y_i and model prediction $\hat{y}_i$). We assume that the residual are approximately asymptotically normal distributed. Given the set of residuals $\mathcal{D} = \{r_i\}_{i=w+1}^{n} = \{y_i - \hat{y}_i\}_{i=1}^{n}$, the generalized extreme studentized deviate test (ESD) [25] is applied to detect statistical outliers in $\mathcal{D}$. The hypotheses of ESD are:

$$H_0 : \text{There are no anomalies in the data.}$$
$$H_a : \text{There are up to } k \text{ percent of anomalies.}$$

In ESD, only an upper bound is required for the suspected number of outliers and not the exact number of outliers. The test is performed in an iterative manner: in each step, the extreme value $r^* = \arg\max_{r_i \in \mathcal{D}} |r_i - \bar{r}|$ with sample mean $\bar{r}$ is checked against the null hypothesis. The test statistic is:

$$R = \frac{\max\{|r_i - \bar{r}| : r_i \in \mathcal{D}\}}{s} = \frac{r^*}{s}, \tag{1}$$

where s is the sample standard deviation. At the significance level α, the test statistic R is tested against the critical value:

$$\lambda = \frac{(n-1)t_{p,n-2}}{n\sqrt{(n-2+t_{p,n-2}^2)}},\ p = 1 - \frac{\alpha}{2n},\ n = |\mathcal{D}|, \tag{2}$$

with n denoting the size of the data set and $t_{p,n-2}$ denoting the $100 \cdot p$ percentage point of the t-distribution with $n-2$ degrees of freedom [25]. The null hypothesis H_0 is rejected when $R > \lambda$. Regardless of the decision on H_0, the current extreme value r^* is removed from $\mathcal{D}$ and the same procedure is repeated on the remaining $n' = n - 1$ data points. The iteration stops when at most k percent of the data points have been removed from the initial $\mathcal{D}$. Outliers are the removed data points on which the null hypothesis is rejected. The ESD procedure is summarized in Alg. 1. In the following subsections, we will give a brief description to the regression models used in our approach.

Algorithm 1 Generalized Extreme Studentized Deviate

```
 1: procedure ESD($k, \mathcal{D}$)    ▷ $k$: maximal percentage of anomalies; $\mathcal{D}$: set of residuals
 2:     $N \leftarrow \lceil k|\mathcal{D}| \rceil,\ n \leftarrow |\mathcal{D}|$
 3:     for $i = 1, 2, \ldots, N$ do
 4:         $\bar{r} \leftarrow \textsc{mean}(\mathcal{D}),\ s \leftarrow \textsc{std}(\mathcal{D})$
 5:         $R_i \leftarrow \max\{|r - \bar{r}| : r \in \mathcal{D}\}/s$
 6:         Compute $\lambda_i$ according to Eq. (2)
 7:         $r_i^* \leftarrow \arg\max\{|r_j - \bar{r}| : r_j \in \mathcal{D}\}$
 8:         $\mathcal{D} \leftarrow \mathcal{D} \setminus \{r_i^*\},\ n \leftarrow n - 1$
 9:     end for
10:     Set $q$ to the last index where $R_i > \lambda_i$ is true
11:     return $\{r_1^*, r_2^*, \ldots, r_q^*\}$
12: end procedure
```

2.1 Random Forest

Random Forest [10] is an ensemble of classification or regression trees [11], where the tree is randomized to reduce the modeling variance as follows: 1) Each tree is trained on bootstrap samples from the data and 2) only a random subset of input features are chosen for the splitting procedure when growing each tree. As for the overall prediction, the predictions from all trees are averaged for regression tasks and a majority vote is taken for classification tasks.

2.2 Support Vector Machine

The support vector machine (SVM) [15] is originally proposed to perform a binary classification task, where the optimal separating (hyper-)plane is obtained by maximizing the distance between sharp linear boundaries of two classes. If the problem is not linearly separable, the well-known kernel trick [8] is applied to map the input into the high dimensional feature space (through the so-called feature map), where the linearly separability is again attained.

2.3 *k*-Nearest Neighbour Regression

k-Nearest Neighbour Regression (kNN regression) [3] is a proximity based algorithm where the target value of a data point is estimated by the average value of its k nearest neighbours. The proximity between data points is determined by some distance metric, e.g., the Euclidean distance which is used in this paper. In addition, a weighting scheme is commonly used in averaging target values of the neighbours and the weight is scaled with respect to the proximity.

2.4 Artificial Neural Networks

Artificial Neural Networks (ANN) are structural machine learning algorithms that are inspired by biological nervous systems [6]. The so-called feedforward neural network model is adopted here. The feedforward neural network [26] consists of multiple layers of artificial neurons which take inputs from the preceding layer and feeds the output of some activation function to the next layer. A feedforward neural network (with more than two layers) is able to approximate any continuous function, according to the Universal Approximation Theorem [16].

2.5 Gaussian Processes

The formal definition of Gaussian processes is that it is a "collection of random variables, any finite number of which have a joint Gaussian distribution" [24]. This Gaussian distribution is completely defined by its mean vector and its covariance matrix, which are calculated based on the training data with applying the kernel trick. However, the prediction is not solemnly based on this joint Gaussian distribution results but also the minimization of the expected loss. Through the covariance matrix this algorithm directly incorporates a uncertainty measure for every prediction.

2.6 ARIMA

Auto Regressive Integrated Moving Average (ARIMA) [9] uses previously observed data (AR) and the moving average (MA) of previous data points to model not directly the output, but the d-th differenced output value (integrated). This is done to derive a stationary time series, which is needed for being able to extrapolate information gained on one section of the time series to the next.

2.7 AdVec

This algorithm developed by Twitter, also commonly referred to as AdVec based on its R command *AnomalyDetectionVector*, decomposes the time series using the seasonal-trend decomposition procedure based on LOESS regression (STL) [14] and applies a form of the generalized ESD on the calculated residuals on a time frame. Instead of taking the mean value the evaluation here uses the median in the ESD test for a more robust estimation against outliers [17].

Table 1. Summary of the Data sets

Data sets		Number of Observations	Value range	Number of Anomalies
Yahoo				
A1 (67)	min	741	0	0
	max	1,461	7,845,760	12
A2 (100)	min	1,421	-2,204	1
	max	1,421	128,420	3
A3 (100)	min	1,680	-7,988	1
	max	1,680	7,006	16
A4 (100)	min	1,680	-6,171	1
	max	1,680	6,324	16
Numenta				
Artificial w. Anomaly (6)	min	4,032	-22	1
	max	4,032	165	1
realAdExchange (6)	min	1,538	0	1
	max	1,643	16	4
realAWSCloudwatch (16)	min	4,032	0	0
	max	4,730	863,964,000	3
realKnownCause (7)	min	1,882	0	2
	max	22,695	39,197	5
realTraffic (5)	min	1,127	0	1
	max	2,500	5,578	4
realTweets (10)	min	15,831	0	2
	max	15,902	13,479	5

2.8 Ensemble Method

To use all aforementioned algorithm as an ensemble, the simple majority vote method is taken, which predicts an anomaly if at least three of the methods introduced in subsections 2.1-2.7 do so.

3 Data Sets

We use two publicly available anomaly detection data sets as benchmark, namely the Yahoo S5 data set [19] and the Numenta data set [20]. These two main data sets are divided into different classes, where each class contains multiple not necessary related time series, which vary in length, value range and anomalies contained as summarized in Table 1. They are briefly described in the following sections.

3.1 Yahoo S5 Data Set

This data set is divided into four classes, named A1, A2, A3 and A4. While A1 consists of 67 real time series from computational services, the other three data sets include each 100 artificially created time series with inserted anomalies and of increasing difficulty. The value range and frequency is not comparable within a class, so that each time series in a class is considered a separate data set. The number of anomalies in each time series is between 0 and 16.

3.2 Numenta Data Set

The Numenta time series benchmark consists of 58 labeled data sets from different domains with multiple anomalies. The domains range from social media chatter to network utilization and also artificially generated data sets. The anomalies are labeled as such and an anomaly window is defined around them. We did not include the artificial time series where no anomalies were present, and excluded one of the network utilization data sets where the labeling was not clear. It is also important to mention that either the root cause for the anomalies in the Numenta benchmark is known, or anomalies were labeled according to "a result of the well-defined NAB labeling procedure" [20]. As in the Yahoo S5 data set each time series in a domain is considered a separate data set with number of anomalies ranging from 0 to 5.

4 Experimental Results

We used standard R packages for all algorithms and the default parameter settings where possible[4].

4.1 Pre-processing Techniques

There are several different approaches for the setup of time series prediction (e.g., see [7]). One approach is to decompose the time series into pattern and features [4]. Therefore some kind of window size needs to be set. This parameter can be optimized based on the precision of a model. But since the goal in this paper is to find unsupervised anomalies and not to create a good prediction model we did not apply these approaches. However, since time series are often incorporated with seasonality, we used the partial auto correlation [9] to automatically determine a window size. We chose a value for the window size which was slightly higher than the highest correlated value at least 11 points away. Because especially in the artificial Yahoo data set some periods lasted longer than 200 points we chose equidistantly 10 points in such a period for the input values of our regression model. However the F1 scores were a lot worse than the ones presented in Table 2. Next to data operations like scaling, detrending, log-transformation and normalization [2,5,29] we conducted another experiment where we computed the first order differences of the time series and used these as inputs. However, this did not improve the F1 scores so no pre-processing was applied for our final results in Chapter 4.

To analyze the sensitivity of the approach based on w, we tested the window sizes $w \in \{5, 10, 15, 20\}$ and found that the results are insensitive to a choice from those four values. However, to provide a consistent learning window, we use $w = 10$ in the following.

[4] More precisely: "ranger" for Random Forests with `mtry` set to 5, "e1071" for SVM, "FNN" for k-nearest neighbours with k equals 10, "nnet" for the neural network with 20 nodes, "kernlab" for Gaussian Processes, "forecast" for ARIMA and `github("twitter/AnomalyDetection")` for AdVec with periodicity of 40, which was found most useful in [27].

4.2 Performance Metrics and Tests

After the regression models were trained and again used to predict the time series, we calculated the residuals $r_i = y_i - \hat{y}_i \; \forall i = 1, ..., n$ and applied the ESD test on them. The ESD test here uses a significance level of $\alpha = 0.05$ to find at most 0.5% of the data set as anomalous. This small ratio of anomalies is the same for all data sets and algorithms. In the AdVec algorithm a modified version of ESD is already encoded, so that it is not reapplied.
In case of the Numenta data set not only punctual anomalies are given, but also an anomaly window. This has an impact on the evaluation: The exact anomaly does not necessarily have to be found; instead an anomaly is considered to be detected if at least one observation in the anomaly window is labeled as an anomaly. Furthermore, multiple points labeled as an anomaly in the same anomaly window do not affect the performance measure, whereas each point which is labeled as an anomaly but is not within the window is considered as a false positive.

The following performance measures are used in our study:

1. True positives (TP): The number of anomalies correctly detected as such.
2. False positives (FP): The number of data points labeled by the underlying algorithm as anomalies even though they are normal instances.
3. False negatives (FN): The number of anomalies that are not detected.
4. Precision: The ratio of correctly labeled anomalies over all anomalies which are detected, i.e.: precision $= \frac{\text{TP}}{\text{TP} + \text{FP}}$.
5. Recall: The ratio of all correctly labeled anomalies over all actual anomalies, i.e.: recall $= \frac{\text{TP}}{\text{TP} + \text{FN}}$.
6. F1 score: The harmonic mean between precision and recall, i.e. F1 $= 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$. This is the most reliable score since maximizing precision and recall are often conflicting goals.

4.3 Results

The results are summarized in Table 2 where the average values of precision, recall and F1 score are shown for all data sets divided into their sub-classes.

For the Numenta benchmark the results are quite satisfactory in terms of the recall measure, implying that almost all anomalies are found. In contrast, the precision values are in general very low and never exceed 0.4, which indicates high rates of false alarms. Consequently, the F1 score is rather low for the Numenta data set. One reason for this observation is posed by the anomaly labels provided by the data set. In Figure 1 a time series from the Numenta data set is shown which has only two labeled anomalies. No algorithm we tested was able to detect those anomalies, but instead false positive detection of peak values can be observed. In case of univariate time series one would assume that anomalies should be understandable. However, without additional information on the nature of the time series it is currently unclear why these anomalies are actually labeled as such, whereas all the clearly visible spikes and peaks are not labeled as anomalies. Therefore, the observed behaviour is actually more or less expected

Table 2. Average performance on the data set groups using the lagged method with window size $w = 10$.

		Algorithm							
Data set	Measure	RF	GP	SVM	*k*NN	ARIMA	AdVec	ANN	Ens
Yahoo									
A1 (67)	P	0.4	**0.48**	0.46	0.44	0.41	0.44	0.43	**0.48**
	R	0.77	0.72	0.76	0.76	0.75	0.48	0.55	**0.79**
	F1	0.52	0.58	0.58	0.56	0.53	0.46	0.49	**0.6**
A2 (100)	P	0.48	0.67	0.82	0.48	0.4	1	**0.84**	0.62
	R	**0.98**	**0.98**	**0.98**	**0.98**	0.97	0.29	0.54	**0.98**
	F1	0.65	0.79	**0.9**	0.65	0.57	0.45	0.66	0.76
A3 (100)	P	0.84	0.86	0.83	0.87	0.66	**1**	0.9	0.89
	R	**0.69**	**0.69**	0.64	0.68	0.54	0.02	0.21	0.67
	F1	0.75	**0.77**	0.72	0.76	0.6	0.03	0.34	0.76
A4 (100)	P	0.69	0.71	0.71	0.68	0.57	0.28	**0.81**	0.77
	R	**0.61**	0.6	**0.61**	0.59	0.49	0.06	0.2	0.6
	F1	0.65	0.65	0.66	0.63	0.53	0.1	0.32	**0.67**
Numenta									
Artificial w.	P	0.04	0.06	0.06	0.06	0.04	0.05	**0.07**	0.05
Anomaly (6)	R	0.83	0.83	**1**	**1**	0.67	0.67	**1**	**1**
	F1	0.08	0.11	0.11	0.11	0.07	0.09	**0.12**	0.1
realAd	P	0.26	0.27	0.28	0.26	0.29	**0.38**	0.24	0.26
Exchange (6)	R	0.71	0.71	0.71	0.71	0.71	0.71	0.71	0.71
	F1	0.38	0.39	0.4	0.38	0.42	**0.5**	0.36	0.38
realAWS	P	0.11	0.11	0.1	**0.12**	**0.12**	0.09	**0.12**	0.11
Cloudwatch	R	**0.93**	0.82	**0.93**	**0.93**	0.89	0.79	**0.93**	**0.93**
(16)	F1	0.19	0.19	0.19	**0.21**	**0.21**	0.17	**0.21**	0.2
realKnown	P	0.06	0.07	0.07	0.05	0.11	0.06	**0.15**	0.07
Cause (7)	R	**0.63**	0.58	0.63	0.58	0.58	0.42	0.37	0.58
	F1	0.11	0.12	0.13	0.1	0.19	0.11	**0.21**	0.13
realTraffic	P	0.22	0.28	0.24	0.28	**0.3**	0.21	0.21	0.23
(5)	R	0.79	0.79	0.79	0.79	**0.93**	0.64	0.71	0.79
	F1	0.34	0.41	0.37	0.41	**0.45**	0.32	0.33	0.35
realTweets	P	0.05	**0.06**	0.05	0.05	**0.06**	**0.06**	**0.06**	0.05
(10)	R	**1**	0.97	0.94	0.97	0.97	0.91	0.97	0.97
	F1	0.1	**0.11**	0.1	0.1	0.1	**0.11**	**0.11**	0.1

from unsupervised anomaly detection algorithms. With additional information, the number of false positives could be reduced if, for example, all peak values above 0.2 should not be considered abnormal.

Concerning the Yahoo data sets, we frequently observe a better balance between precision and recall than for the Numenta data sets and, consequently, a much higher value of the F1 score.

We illustrate an exemplary time series from the Yahoo benchmark in Figure 2 with the corresponding anomalies marked in red but without anomaly windows. This artificially created time series reveals a general problem of our unsupervised anomaly detection method. The anomaly detection algorithms learn the overall statistics of a complete dataset, and anomalies are detected by inspecting the difference between prediction and actually observed value. Therefore, a change in the average statistics and of the overall scale of the time-series is not well-captured by the detection method and the anomalies in the first third of the dataset are masked by the last part, where the amplitude increased. As a result, no method used here can capture the anomalies within the initial low amplitude signal. Such cases would benefit strongly from employing an online algorithm where, for example sliding analysis windows or forgetting factors are used to adjust to qualitative changes of the time series.

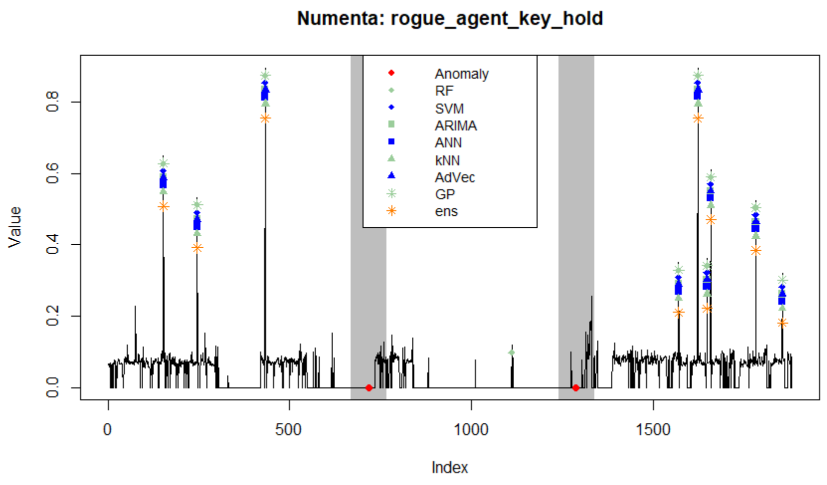

Fig. 1. A Numenta time series with two anomaly windows marked in grey, where the two red dots indicate the actual anomalies. None of the algorithms tested has been able to identify the true anomalies, but all marked the high peak values as anomalies. Indeed, it seems difficult to derive the characteristics of the two labeled anomalies from the underlying time series.

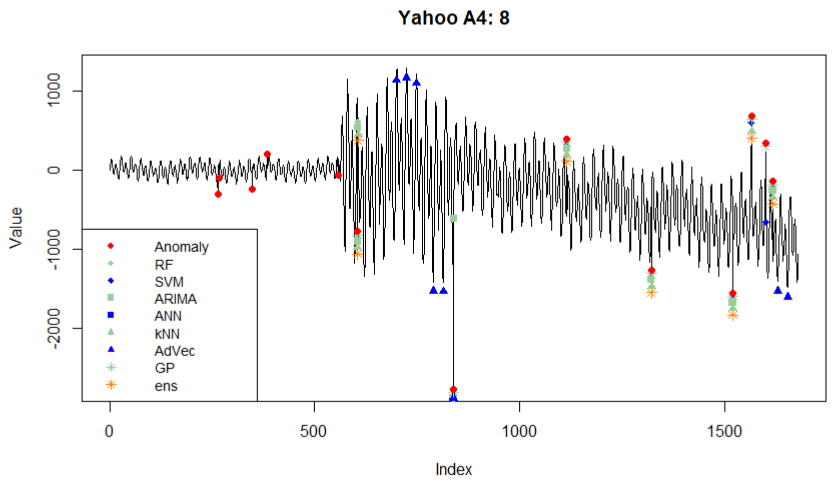

Fig. 2. Yahoo time series A4: 8, with 13 labeled true anomalies (red dots). None of the algorithms tested can correctly identify the first five anomalies, while the next 8 are correctly identified by gaussian processes, and 7 by four other methods. AdVec generates seven false positives.

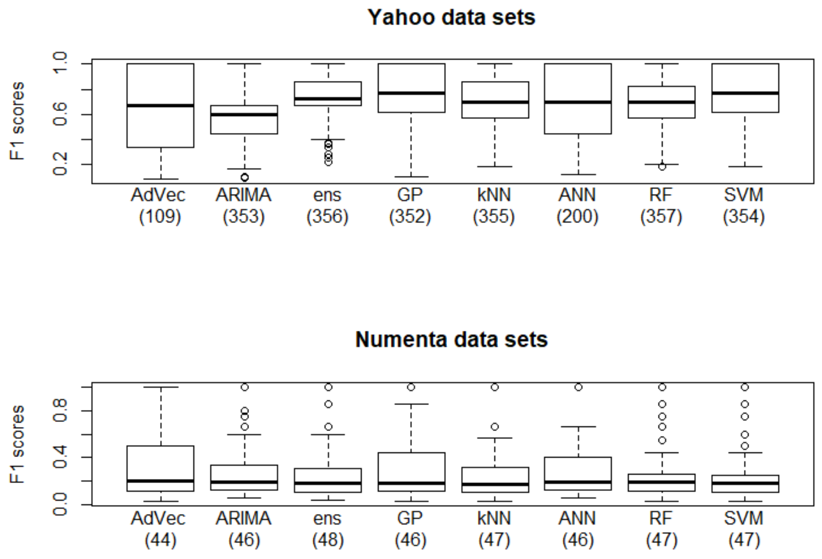

Fig. 3. F1 scores represented as boxplots for the corresponding algorithm and divided by data set affiliation. The thick horizontal line indicates the median and the box corresponds to the 25% and 75% quantile. The number on the x-axis corresponds to the number of time series where the F1 score was successfully calculated. Unsuccessful calculations are those where the algorithm could not calculate the F1 score because of a division by 0, i.e. if no true positives were found.

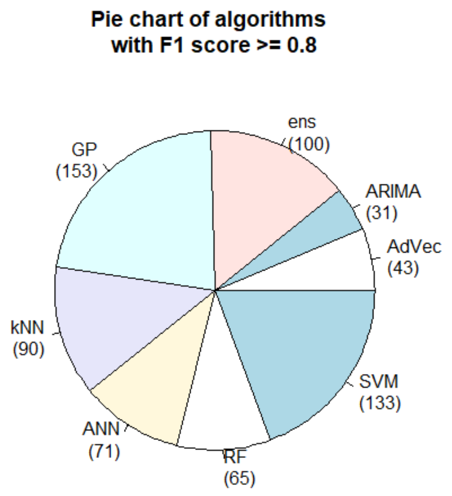

Fig. 4. Distribution of algorithms with the highest F1 score for scores larger than 0.8. The number under each algorithm's name is the number of time series where this algorithm could reach the maximal F1 score. Because multiple algorithms can have the same maximal F1 score for one time series the cumulative numbers exceed the number of time series.

Table 3. Mean correlation based on block wise sampled test data.

	Algorithm						
Data set	RF	GP	SVM	*k*NN	ARIMA	AdVec	ANN
Yahoo							
A1 (67)	0.75	0.60	0.70	0.74	**0.82**	0.80	0.40
A2 (100)	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	0.56
A3 (100)	**0.97**	**0.97**	**0.97**	**0.97**	**0.97**	0.95	0.71
A4 (100)	**0.97**	0.96	**0.97**	0.96	0.96	0.94	0.75
Numenta							
Artificial w. Anomaly (6)	0.82	0.76	0.67	0.80	**0.90**	0.78	0.81
realAdExchange (6)	**0.64**	0.60	0.28	0.26	0.29	0.38	0.24
realAWSCloudwatch (16)	0.11	0.11	0.1	**0.12**	0.12	0.09	**0.12**
realKnownCause (7)	0.06	0.07	0.07	0.05	0.11	0.06	**0.15**
realTraffic (5)	0.22	0.28	0.24	0.28	**0.3**	0.21	0.21
realTweets (10)	0.05	0.06	0.05	0.05	**0.06**	**0.06**	**0.06**

Figure 3 shows the statistics of the F1 scores for each algorithm over each time series. Generally, there is a substantial variation in the F1 scores for all methods and data set. As already observed, the performance on the Yahoo data set is generally acceptable where the majority results are located in the upper half of performance scores, i.e. between 0.5 and 1. For AdVec the bulk of the results is distributed over the complete score scale and an anomaly label could only be calculated for about one third of time series due to no anomaly detections at all. The minimum (i.e. worst) F1 score generated by all algorithms is between 0 and 0.2, but at the same time all methods reach once or multiple times a score very close to 1. While for the Numenta data set the minimal and maximal F1 score are comparable, the general performance is much worse where the median values are found around an F1 score of about 0.2. This indicates that many anomalies labeled in time series of the Numenta data set are much more difficult to find.

Even though the prediction accuracy is not the main interest here, we analyzed how well the algorithms perform on the given data sets. In order to divide the time series in training and test set, we choose a block-wise sampling where in total 20% of the data was sampled in 10 equidistantly distributed blocks to function as the test data. (We decided against using 80% of the first part of the data for training and the remaining last 20 % to test, since the time series contain trends, concept drifts, different seasonality pattern and anomalies at the end of the observation period. We also did not use random sampling, since for time series data this leads to no reliable prediction accuracy because of the natural link between two consecutive points is lost.) Because of the strong variation in the range of values (cf. Table 1), the averaged correlation was used instead of the average mean squared error. In Table 3 we observe, that the prediction capability of models trained on the Yahoo S5 data set are better than for those in the Numenta data set. Interestingly, a good prediction capacity does not necessarily be connected with the best anomaly score, compare Numentas realAdExchange, where random forests provided the highest correlation but AdVec the best F1 score.

From the above presented results we can conclude that none of the studied anomaly detection methods significantly outperforms any other approach. All algorithms performed similarly, except for AdVec where the F1 score was often not calculable (due to zero detected anomalies) and showed more variation if calculable. The reason for this is very likely found in the target application scenario for which AdVec was designed and the implicit assumptions about the data made there. Presumable, with fine tuning the periodicity hyperparameter according to each time series AdVec would perform better. The simple ensemble method we applied is a bit more robust against the variation of an individual algorithm and thus can achieve good overall performance. Analyzing the frequencies with which the algorithms achieve F1 scores equal or higher than 0.8 in Figure 4, we find that Gaussian processes and support vector machine provide the most high F1 scores.

5 Conclusions and Future Work

A total of seven machine learning algorithms and one simple ensemble method were compared on a large number of univariate time series data sets commonly used in anomaly detection. The evaluation criteria used here were precision, recall and F1 score. A central finding of this work is that all methods perform comparably, with a slight favor for gaussian processes, support vector machine and the ensemble method. On the Numenta data sets, all methods show reasonably good recall performance while they all perform poorly with respect to the precision and consequently also the F1 measure, which is due to a high rate of false detections. For the Yahoo data set the precision and recall measures are much more balanced and therefore the F1 scores are usually much larger. Our results show a tendency that an ensemble method will yield a more constant performance than an individual algorithm. This is intuitive as it reflects the usual insight that it is beneficial to combine many weak learners into a strong learner. Put differently, individual algorithms failing on some time series will not affect the average ensemble performance much as long as there is no systematic reason for many algorithms to fail simultaneously. In the current approach, we did not do any hyperparameter tuning and also did not use any a priori information about the types and characteristics of the data sets. This was done to achieve an unbiased comparison of the algorithms for situations where not much knowledge about the data is available and uniformed analysis of real-world data is targeted. However, it can be expected, that the performance will improve when a priori information about the data is utilized. There are many routes to improving the performance of the anomaly detection approach presented in this paper. One such improvement could be realized by utilizing an online approach so that the algorithms are not trained globally on all available data. Instead, the algorithm could adjust constantly to the recently observed data by, for example, using a sliding window approach or introducing a forgetting factor. With this, the algorithms would be able to adapt to seasonality, concept drift or other global changes in the time series. Additionally, more advanced data pre-processing and feature detection could be employed. Especially the development of automated data pre-processing techniques should further be investigated. The approach presented here can be applied without intensive data processing to get anomalies flagged in an industrial use case where many different time series are present and no labeling information exists. As in [28], these anomaly suggestions are to be verified by the process expert and pre-studies performed on a network use case nurture the hope that this investigation method contributes to an increased process understanding. These options are left for further research studies.

References

1. Ahmad, S., Purdy, S.: Real-time anomaly detection for streaming analytics. CoRR **abs/1607.02480** (2016), http://arxiv.org/abs/1607.02480
2. Ahmed, N.K., Atiya, A.F., Gayar, N.E., El-Shishiny, H.: An empirical comparison of machine learning models for time series forecasting. Econometric Reviews **29**(5-6), 594–621 (2010)
3. Altman, N.S.: An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician **46**(3), 175–185 (1992)
4. Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.: The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances. Data Min. Knowl. Discov. **31**(3), 606–660 (May 2017). https://doi.org/10.1007/s10618-016-0483-9, https://doi.org/10.1007/s10618-016-0483-9
5. Balkin, S.D., Ord, J.K.: Automatic neural network modeling for univariate time series. International Journal of Forecasting **16**(4), 509–515 (2000)
6. Bishop, C.M.: Pattern recognition and machine learning. Information Science and Statistics, Springer, New York (2006). https://doi.org/10.1007/978-0-387-45528-0, https://doi.org/10.1007/978-0-387-45528-0
7. Bontempi, G., Ben Taieb, S., Le Borgne, Y.A.: Machine learning strategies for time series forecasting. In: Aufaure, M.A., Zimányi, E. (eds.) Business Intelligence: Second European Summer School, eBISS 2012, Brussels, Belgium, July 15-21, 2012, Tutorial Lectures, pp. 62–77. Springer Berlin Heidelberg, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36318-4_3, https://doi.org/10.1007/978-3-642-36318-4_3
8. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on Computational learning theory. pp. 144–152. ACM (1992)
9. Box, G.E., Jenkins, G.M., Reinsel, G.C., Ljung, G.M.: Time series analysis: forecasting and control. John Wiley & Sons (2015)
10. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
11. Breiman, L., et al.: Classification and Regression Trees. Chapman & Hall, New York (1984)
12. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: Lof: Identifying density-based local outliers. SIGMOD Rec. **29**(2), 93–104 (May 2000). https://doi.org/10.1145/335191.335388, http://doi.acm.org/10.1145/335191.335388
13. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) **41**(3), 15 (2009)
14. Cleveland, R.B., Cleveland, W.S., McRae, J.E., Terpenning, I.: Stl: A seasonal-trend decomposition procedure based on loess. Journal of Official Statistics **6**(1), 3–73 (1990)

15. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20**(3), 273–297 (Sep 1995). https://doi.org/10.1007/BF00994018, https://doi.org/10.1007/BF00994018
16. Csáji, B.C.: Approximation with artificial neural networks. Faculty of Sciences, Etvs Lornd University, Hungary **24**, 48 (2001)
17. Hochenbaum, J., Vallis, O.S., Kejariwal, A.: Automatic anomaly detection in the cloud via statistical learning. arXiv preprint arXiv:1704.07706 (2017)
18. Kriegel, H.P., Kröger, P., Schubert, E., Zimek, A.: Loop: local outlier probabilities. In: CIKM. pp. 1649–1652. ACM (2009), "http://dblp.uni-trier.de/db/conf/cikm/cikm2009.html#KriegelKSZ09"
19. Laptev, N., Amizadeh, S.: Yahoo anomaly detection dataset S5 (2015), available under https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70
20. Lavin, A., Ahmad, S.: Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark. In: Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on. pp. 38–44. IEEE (2015)
21. Marteau, P.F., Soheily-Khah, S., Béchet, N.: Hybrid isolation forest - application to intrusion detection. CoRR **abs/1705.03800** (2017), http://arxiv.org/abs/1705.03800
22. Muller, K., Smoła, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., Vapnik, V.: Predicting time series with support vector machines. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 1327, pp. 999–1004. Springer Verlag (1997)
23. Paffenroth, R., Kay, K., Servi, L.: Robust PCA for anomaly detection in cyber networks. CoRR **abs/1801.01571** (2018), http://arxiv.org/abs/1801.01571
24. Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. Adaptive Computation and Machine Learning, MIT Press, Cambridge, MA, USA (Jan 2006)
25. Rosner, B.: Percentage points for a generalized esd many-outlier procedure. Technometrics **25**(2), 165–172 (1983)
26. Schmidhuber, J.: Deep learning in neural networks: An overview. Neural Networks **61**, 85 – 117 (2015). https://doi.org/https://doi.org/10.1016/j.neunet.2014.09.003, http://www.sciencedirect.com/science/article/pii/S0893608014002135
27. Thill, M., Konen, W., Bäck, T.: Online anomaly detection on the webscope s5 dataset: A comparative study. In: 2017 Evolving and Adaptive Intelligent Systems (EAIS). pp. 1–8 (May 2017). https://doi.org/10.1109/EAIS.2017.7954844
28. Vercruyssen, V., Meert, W., Verbruggen, G., Maes, K., Baumer, R., Davis, J.: Semi-supervised anomaly detection with an application to water analytics. pp. 527–536 (11 2018). https://doi.org/10.1109/ICDM.2018.00068
29. Zhang, G.P., Qi, M.: Neural network forecasting for seasonal and trend time series. European Journal of Operational Research **160**(2), 501–514 (2005), https://EconPapers.repec.org/RePEc:eee:ejores:v:160:y:2005:i:2:p:501-514