Machine learning and deep learning approaches for multivariate time series prediction and anomaly detection

Thill, M.

Chapter 3

Related Work

In this chapter, we shortly describe several state-of-the-art anomaly detection algorithms that we use in this work for comparison purposes. Additionally, we give an overview of the current relevant state of the art in the field of time series anomaly detection. Since this field is very wide, the algorithms and benchmarks discussed in this chapter are in no way intended to represent a comprehensive list. For a detailed overview and classification of the different algorithms and datasets, the reader is referred to general anomaly (novelty) detection survey papers [26, 64, 131, 135, 109, 110, 23, 128] or survey papers with the focus on time series [31, 8, 16, 55]. Additional, more specific related work is also presented in the following chapters.

3.1 Algorithms used for Comparison Purposes

This section describes all time series anomaly detection algorithms used in this thesis for comparison purposes. Our own contributions will be described in the course of the thesis: Chapter 4 – 7 introduce SORAD, DWT-MLEAD (offline & online variant), LSTM-AD, and TCN-AE, respectively.

3.1.1 ADVec

Twitter's ADVec algorithm [173] is a robust online anomaly detection algorithm designed to detect local and global anomalies in time series with seasonal/periodic behavior and underlying trend. ADVec uses a method called Seasonal Hybrid ESD (S-H-ESD), which is based on the Generalized ESD (extreme Studentized deviate) test [141], combined with robust statistical approaches and piecewise approximation. It is available as open-source R package "AnomalyDetection" from Github.[1] The algorithm requires three main parameters: The first parameter α describes the level of statistical significance with which to accept or reject anomalies. As in the other algorithms, this parameter can be interpreted as an anomaly threshold. ADVec requires a second parameter, a period-length. The third parameter, $\max_{anoms}$, determines the maximum number of anomalies that the algorithm will

[1] `http://github.com/twitter/AnomalyDetection`

detect as a percentage of the data. We found that the setting of the parameter $\max_{anoms}$ is crucial (but not difficult to select) for the performance of ADVec.

3.1.2 NuPIC

Numenta's online anomaly detection algorithm NuPIC [160] is based on the hierarchical temporal memory (HTM) model [49] which is biologically inspired by the neocortex of the brain. NuPIC is also open-source and can be obtained from GitHub[2]. To verify the correctness of our NuPIC installation, we applied it to the Numenta Anomaly Benchmark (NAB) ([89], section 2.3.2) using the standard parameter settings and confirmed that we could exactly reproduce the results[3] published in [89]. The anomaly detection algorithm behind NuPIC, similarly to other approaches, is based on the assumption [160, 89] that time series are predictable to a certain extent. At each time step t, NuPIC performs several predictions for time step $t+1$. These predictions are compared with the time series's actual value at $t+1$ and based on the prediction errors, an anomaly score is formed. NuPIC also tracks and adapts the estimated mean and variance for the anomaly scores in an online fashion and can thus handle dynamically changing behavior to some extent. Furthermore, it is supposed to work on a large variety of datasets without manual parameter tweaking [89]. Although the parameters can be tuned with an internal swarming tool [3], a tool to aid automatic parameter search for a given dataset, we decided to use the standard parameter settings recommended in [89] for all investigated problems. For some problems, the time-expensive tuning process is not feasible, and for the other problems, we found that the results were very similar, so we list the results for the standard parameter settings in all chapters. NuPIC outputs an anomaly likelihood in the interval [0,1] for each time series point, which is suitably thresholded to control the algorithm's sensitivity.

3.1.3 LSTM-ED

LSTM-ED [108] attempts to reconstruct sub-sequences of time series in order to detect anomalies. It addresses the problem that many time series are not predictable (in a sense that it is impossible to accurately forecast the time series's next values). Internally, it uses an encoder-decoder LSTM [63, 54, 62] network, which takes sub-sequences from the time series and encodes them into vectors of fixed length and, subsequently, attempts to reconstruct the whole input sequence from the encoded vector. For each reconstructed point, a reconstruction error is computed. For the reconstruction errors, a Gaussian distribution is estimated, and the probability density of each point is used as anomaly score. The algorithm's main parameters are batch size B, the number of training epochs n_{epochs}, sequence length T_{train}, hidden size $h = 100$ and $\%_{Gaussian}$, which specifies the fraction of the data

[2] `https://github.com/numenta/nupic`

[3] The result files can be obtained from `https://github.com/numenta/NAB/tree/master/results/numenta`.

used to estimate a Gaussian distribution for the anomaly detection task. Both encoder and decoder use a stacked LSTM network with two layers.

3.1.4 DNN-AE

DNN-AE [46] is similar to LSTM-ED in that it takes short sequences from a time series and attempts to encode and reconstruct these. However, it is a conventional deep autoencoder architecture improving an earlier approach based on replicator neural networks [58]. Contrary to the replicator neural network approach, DNN-AE uses a significantly deeper architecture, trains its weights with the more powerful ADAM optimizer [82], and replaces the step-wise (staircase) activation function[4] at the bottle-neck layer with $\tanh(\cdot)$. The algorithm requires several parameters similar to LSTM-ED: batch size B, number of training epochs n_{epochs}, sequence length T_{train} and a hidden size of h for the bottle neck (which results in a compression factor of T_{train}/h for each sequence) and $\%_{Gaussian}$ (same as for LSTM-ED). The number of densely connected layers L depends on T_{train} and h: The number of units in all hidden layers (except for the bottle neck) are powers of two which are smaller than T_{train} and h. For example, for $T_{\text{train}} = 50$ and $h = 10$ we have $L = 6$ layers and the number of units for the individual layers is $(32, 16, 10, 16, 32, 50)$.

3.2 Other Anomaly Detection Algorithms

In recent years much effort was put into the design of time series anomaly detection algorithms, and researchers proposed many new methods. In this section, we give an overview of different approaches. Due to the great variety in this field, this overview is not exhaustive.

3.2.1 Online Algorithms

Next to the already introduced algorithms ADVec [173] and NuPIC [160, 49] many very different approaches exist for online anomaly detection in time series, for example: Yao et al. [181] present a simple method, called Segmented Sequence Analysis (SSA), which uses the similarity of piecewise linear representations of univariate time series and a reference model to detect anomalous behavior in sensor data. Ma & Perkins derived an online support vector regression (SVR) training algorithm [104] which they use to predict time series and detect anomalies (novelties) based on the prediction error [102]. In [175], Wang et al. introduce an online algorithm based on two statistical approaches, the Tuckey method, and the relative entropy statistic. Their algorithm is designed to detect anomalies in large-scale cloud services. Ahmed et al. [4] use the kernel recursive least squares (KRLS) algorithm to maintain a relatively small dictionary of feature vectors forming a cluster in a high-dimensional feature space that approximately describes the normal behavior of traffic measurements in a

[4]We also experimented with the step-wise activation function introduced in [58] but could not observe any improvements over using $\tanh(\cdot)$.

network. The maintained dictionary can adapt itself over time and detects anomalous network traffic by computing the distance of new data points to the normal cluster. Talagala et al. [157] present an online algorithm for univariate time series based on extreme value theory and a kernel density estimation. In [158], an adaptable approach for streaming data based on so-called half-space (HS) trees is proposed. Wei et al. developed an online approach based on symbolic time series representations (Symbolic Aggregate approXimation, SAX [94]) which can – according to its authors – be applied to a wide range of applications without domain-specific customization.

We did not find many online-adaptable DL algorithms for time series anomaly detection in the literature: Saurav et al. [146] introduce an online DL model for (multivariate) time series. A stacked GRU (gated recurrent unit) network is used for multistep ahead prediction, and the mean of the prediction errors is used as anomaly score and is compared to a threshold in order to detect anomalies. The learning algorithm is based on a stochastic gradient descent (SGD) approach with an anomaly-score-dependent learning rate, which can perform the backpropagation through time (BPTT) in a fully online fashion.

3.2.2 Deep Learning Approaches

Although some DL algorithms exist, which can be used in an online-adaptable fashion [146], we found that most DL anomaly detection algorithms for time series are trained offline. Due to the temporal nature of the data, common building blocks for DL algorithms are convolutional neural networks or recurrent neural networks (RNNs) such as the long short-term memory (LSTM) [63] or gated recurrent units [29], which are generally applicable to multivariate time series. In some cases also regular fully-connected neural networks are used. Most algorithms that we found in the literature can be classified either as prediction-based or reconstruction-based or are combinations of both.

3.2.2.1 Prediction-based DL Algorithms

The most common approach is to train a DL model to perform a multistep-ahead prediction (forecasting) and compare the predictions to the observed values. Typically, the prediction errors are used as an indicator for anomalous behavior. Although these approaches are trained in a (self-) supervised fashion to predict the time series, they are still considered unsupervised as long as the actual anomaly labels are not used. Malhotra et al. [107] describe such an anomaly detection algorithm based on LSTMs. It predicts over several horizons and estimates a multivariate Gaussian distribution for the prediction errors, and uses the probability density function of the estimated Gaussian as anomaly score. [28, 45, 44, 67, 18, 51] are similar to and partially based on [107], but apply their algorithms to different problems. Partially, also the anomaly thresholds are computed in different ways, for example, simply with the mean squared error (MSE) [45, 44], or using exponentially-weighted error averages [67]. However, most of these models ([107, 28, 45, 44, 67, 18]) are only trained with anomaly-free data and partially also require that the time series data is

manually split into training, validation, and test sets (which requires labeled data or expert knowledge). In chapter 6 (based on [161]), we present a prediction-based LSTM model which can be trained with contaminated data (time series with anomalies) and extends [107] in several aspects.

Instead of LSTMs or GRUs, in [61], He & Zhao use a temporal convolutional network (TCN) [13] for predicting time series, while Munir et al. [115] use regular CNNs. Zhu & Laptev [187] present an algorithm that is a prediction-based approach but uses a reconstruction-based approach (described in the following section) for pre-training: during the pre-training phase, an LSTM encoder-decoder network is trained to extract useful embeddings from a sequence. Subsequently, the decoder is discarded. The encoder network generates features for the prediction network (a multi-layer densely connected network), predicting the next few time-series steps. Additionally, the authors describe an approach to assess the prediction uncertainty, which can help to detect anomalies.

3.2.2.2 Reconstruction-based Algorithms

Other popular approaches for time series anomaly detection are based on compressing and reconstructing time series (or segments thereof) and detecting anomalies using the reconstruction error (or reconstruction probabilities). Reconstruction-based algorithms can be applied when a time series contains normal patterns which are inherently unpredictable. Generally, they are also applicable to predictable time series. The previously introduced DNN-AE ([58], Section 3.1.4) and LSTM-ED ([108], Section 3.1.3) are examples for such approaches. Similarly to [58], other early approaches apply regular (deep) autoencoders (AE) to windows of fixed length and slide the AE over the whole time series. Although relatively simple, these approaches demonstrate their effectiveness in various applications [144, 58, 35]. Oh & Yun [123] present an autoencoder architecture based on CNNs, which is trained to encode and reconstruct segments of a spectrogram. If the reconstruction SSE (sum of squared errors) of a segment lies above a predefined threshold, the corresponding segment is flagged as anomalous. Kieu et al. [80] use an autoencoder architecture based on 2D CNNs and another LSTM encoder-decoder approach similar to [108]. Additionally, they enrich time series with additional features before passing them to the autoencoder. In [79], Kieu et al. present an autoencoder approach based on an ensemble of sparse recurrent neural networks. Zong et al. present with DAGMM an architecture [188] where the parameters of a deep autoencoder and a Gaussian mixture model are simultaneously learned during training. Zhang et al. [184] construct so-called signature matrices of time series segments, capturing the correlations between different dimensions of the time series, and subsequently use an attention-based convolutional LSTM (convLSTM) autoencoder to encode and decode the signature matrices. The residual signature matrices are used as an indicator for anomalous behavior.

Variational approaches Variational autoencoders [83] have a wide range of applications. In recent years, they are also becoming increasingly popular in the field of (time

series) anomaly detection. Generally, variational anomaly detection approaches attempt to compute reconstruction probabilities instead of reconstruction errors [9]. Pereira & Silveira [133] propose a variational Bi-LSTM autoencoder with variational self-attention (VSAM), which takes short segments as input. But instead of reconstructing the original points of the input sequence, the decoder attempts to compute the reconstruction probability of each point by estimating the parameters of a Laplace distribution (again, for each point of the sequence). Given the parameters of the Laplace distribution, the probability (density) of the corresponding input point can be used as anomaly score (here, lower values indicate a larger degree of abnormality). The authors apply their VSAM algorithm to univariate solar photovoltaic generation time series. In later work, they apply similar approaches to ECG data [132, 134]. The Donut algorithm [180] uses sliding windows and applies a variational autoencoder to windows of fixed length. The algorithm is studied on a dataset from a large internet company and outperforms several baseline algorithms. The dataset is not publicly available. Su et al. propose the OmniAnomaly algorithm [153], which is based on stochastic recurrent neural networks and is designed to encode and reconstruct short sub-sequences of a multivariate time series. Similar to most other approaches, the authors use the reconstruction probability as anomaly score. Additionally, they present an approach for an automatic threshold selection. Other similar variational anomaly detection algorithms were introduced by Park et al. [129] (LSTM variational autoencoder) and Su et al. [154] (echo state conditional variational autoencoder).

GAN-based approaches In recent years, the generative adversarial network (GAN), invented by Goodfellow et al. [53], is being applied to anomaly detection tasks [39]. For time series anomaly detection, often encoder-decoder architectures are used for the generator network [186, 71]. In other approaches, the original sequence is reconstructed from an incrementally updated latent space representation (the gradients of an error function are used to improve an initially random latent space vector iteratively) [91, 92]. Most GAN-based approaches compute the anomaly score for a given sample as a weighted average of a reconstruction loss and a discriminator loss [91, 92, 71]. In [186], the authors only consider the reconstruction loss for the anomaly score. Remarkably, all GAN-based approaches that we studied [186, 71, 91, 92] require that the training data is exclusively normal and does not contain any anomalous sequences.

3.2.2.3 Other relevant Algorithms

The HOT SAX algorithm [77, 78] by Keogh et al. is based on Symbolic Aggregate ApproXimation (SAX) [94, 95] and is designed to detect time series discords, i.e., anomalous subsequences in a longer time series. Other worth-mentioning approaches are based on one-class classification (e.g., using support vector machines [103] or using neural networks [24, 149]), artifical immune-system approaches [33, 34], or energy based models [183]. Notable examples from industry – next to ADVec [173] and NuPIC [160] – are Yahoo's EGADS [88], LinkedIn's luminol [97] and Expedia.com's adaptive alerting [43].

3.3 Benchmarks

As described before in Section 2.3, we use Yahoo's Webscope S5 dataset, the Numenta Anomaly Benchmark (NAB) [89], the Mackey-Glass Anomaly Benchmark [167, 166] and the MIT-BIH Arrhythmia database [52, 112, 113] in our work for benchmarking and comparison purpoes.

Apart from these four datasets, other benchmarks can be found in the literature: Filonov et al. generated the synthetic Gasoil Heating Loop (GHL) dataset [45], using a model of a real gasoil plant to simulate hacker attacks. During the attacks, the simulated intruders adjust the setpoints of various process variables. In order to prevent a fault of the system, an anomaly detection algorithm has to detect the unauthorized changes. In total, the benchmark contains 49 19-dimensional time series. One time series contains only normal behavior and is used as the training data. However, many anomalies can be detected by applying a simple threshold to one of the time series' dimensions. A similar benchmark [44] uses the Tennessee Eastman Process (TEP) with simulated cyberattacks to generate industrial multivariate (59-dimensional) time series. In [20], a benchmark containing 276 univariate time series is introduced. The time series are taken from real-world sensor data (such as temperature or light). However, the authors only inserted artificial anomalies. Each time series has a length between 2000 and 18 000 data points and contains 5 to 23 anomalies of various types (random, malfunction, bias, drift, polynomial drift, and combinations). Additionally, a Java program is available to generate new anomalous time series. However, we did not use this benchmark for our work as the anomalies were artificial and partially appeared to be too trivial. Other popular time series anomaly benchmarks are the Soil Moisture Active Passive satellite (SMAP) and Mars Science Laboratory rover (MSL) data by NASA [67] and the Server Machine Dataset (SMD) [153]. A more recent dataset is Skoltech's anomaly benchmark (SKAB) [75], which currently contains more than 30 multivariate time series. Each time series contains about 1 200 points. The data was collected in a testbed consisting of a water circulation system with eight different sensors. In [18], Bontemps et al. describe how they converted the KDD Cup 1999 dataset [76], a network instrusion detection benchmark (initially designed for point-based anomaly detectors), into a time series version.

In our work, we mostly use the standard metrics precision, recall, and F_1-score to compare the performance of algorithms. Additionally, we use precision-recall curves to compare algorithms over a large range of anomaly thresholds. More advanced – but less commonly used – evaluation metrics for time series are introduced in [159] and [89].