Data Analytics
Stiftung Universität Hildesheim
Marienburger Platz 22
31141 Hildesheim
Prof. Dr. Dr. Lars Schmidt-Thieme

Thesis
Unsupervised Real-Time Time-Series Anomaly Detection

Abdul Rehman Liaqat

271336, Liaqat@uni-hidesheim.de

Abstract

Anomaly detection is a crucial task for machine learning due to wide-spread usage and type. In particular, it is worth noting that most data arising in industrial setups are of a streaming nature, thus restricting the range of standard anomaly detection tools. This thesis will identify the potential approaches to learn the identification of abnormal behavior from large-scale streaming data. An empirical comparison of state-of-the-art methods will to be extended by a novel technical contribution. In this thesis, the focus is particularly on streaming time-series Anomaly Detection which changes in nature with time and novel contribution will especially try to target this dynamic nature of time-series.

Contents

1 Introduction 10 pages

1. Usage of streaming data
2. Usage of anomaly detection in general
3. Software2.0 development and anomaly detection role
4. Usage of anomaly detection in streaming data
5. problems in the concept and brief intro to our innovation

1.1 Usage of streaming data

With the increase in networking of objects, amount of data being generated is increasing. A big chunk of this data involves streaming data. Streaming data can be defined as data being generated continuously or with a continuous time interval. Statistically, IoT data has grow from 4.4ZB (Zeta Bytes) to 44.4ZB with 50 billions devices. These devices include a small temperature sensor installed in a room to ECG data to satellite communication data. Even videos are kind of streaming data since they include data points (frames) separated by a constant interval.

1.2 Usage of anomaly detection in general

An anomaly is defined as significantly different data point from other data points. The difference is dependent upon the at least two things. One is the data point being an outlier in general among all data points available. Second is that the data point value doens't make sense in the sequence data is being generated hence it is possible for new data point to be extremely different from all previous ones but still make sense to have in the sequence and vice versa. Detecting an anomaly helps in identifying abnormal behavior of a process with potentially useful information.

1.3 Software2.0 development and anomaly detection role

As described in (reference name Software2), going forward software development especially machine learning based software development will be divided into two parts. One part is the preparation of the data and the second part is the design and optimization of algorithms. Both of these steps are used iteratively. Interestingly, as described in the (reference name Software2), more and more time is spent on accumulating, massaging and cleaning datasets. One major part of this process is outlier detection, anomaly detection and novelty detection. By detection of anomalies, it will become sufficiently easy for human expert to find the one data point in a heap of millions data points and then decide whether to remove that data or get more of the same data points. Need of an anomaly detection algorithm to identify faulty devices or prevent potential system failure before it happens. Thus the anomaly detection algorithm should be generic (unsupervised), be ready to predict on live data and be able to consider the important past data points (temporal and spatial pattern detection).

1.4 Usage of anomaly detection in streaming data

1.5 problems in the concept and brief intro to our innovation

Now to define the problem officialy, time series can be defined as an ordered sequence of values of a variable at equally spaced time intervals. It can be a variable or group of variables. For example a heat sensor will be producing one time series which is why it is called univariate time series. If we also consider the output of humidity sensor and noise sensor then the group of these univariate time series will be called multivariate time series. Usually techniques applied on univariate time series can also be used on multivariate after some modification. That's why we will be focusing on univariate time series only in this research thesis.

A streaming data or streaming time series has some properties specialized to it. A univariate streaming time series will have following properties:

1. Every new data point x_t will be following by a new data point x_{t+1}
2. The time difference between $t-(t-1)*$ and $(t+1)-t$ is approximately the same
3. There is no known ending time of the stream hence making the streaming data a major source of big data. With this requirement it becomes important to identify important part or time range of the series to be used for the processing.

Moving on there are some unique concepts related to uni-variate time series which will be helpful in the understanding of the research problem. Starting with stationary and non-stationary time series. A stationary time series is the one whose statistical properties are constant over time. These statistical properties include mean, variance and auto-correlation of the time series. Similarly with the change in one or more of these statistical properties, non-stationary time series comes into being. These non-stationary time series are also named as Concept drift. Concept drift is a terminology related to time series data specifically which is why we will be using the term Concept drift instead of non-stationary time series in future. There is also an intuitive meaning behind the name of Concept drift.

Considering that there is an underlying probability distribution in every time series and this underlying distribution is named Concept of time series such as

$$Concept = P_t(X) \tag{1}$$

The the concept drift occurs between two times t and u when the distribution or concept changes as

$$P_t(X) \neq P_u(X) \tag{2}$$

And the new data points are generated from the second distribution $p_u(X)$ onward rather then the original $P_t(X)$ for a length of time T. This time T is decided by the domain knowledge and frequency of data points. As in real life systems it is highly likely that there will be a change in the concept of the time-series hence happening of concept drift is very likely and possible. For example, there will be a concept drift in the time-series output of a heat sensor measuring the temperature of a motor of exhaust fan and the motor of the fan is changed. Thus it is necessary for our system to first recognize the concept drift. If there exists a concept drift then the model should adapt itself swiftly to avoid false positives. Similarly if there is not a concept drift then the model should not give much learning weight to these readings. Thus the model adaption according to presence of concept drift is necessary.

Time series data is the output of a streaming data from multiple devices which makes it close to impossible to label it specially for anomalies. Also due to concept drift a regular labeling needed. Since it is very difficult to label streaming time series data to detect anomalies it makes sense to do it in an unsupervised fashion. With unsupervised anomaly detection we don't need any previous labeled data and the system developed will be very generic. An anomaly is defined as a data point X_t or set of data points $X_{t-l:t}$ for which either of following is true $\gg\ll$ Example of each anomaly

1. Probability of point X_t belonging to a probability distribution ϕ is very less then the probability of previous data points X_n where $n \leq t-1$. In other words:

$$P(X_t) \in \phi \ll P(X_n) \in \phi \tag{3}$$

   This type of anomaly is called point anomaly. Here ϕ is the probability distribution generating all the data points.

2. Probability of points $X_{t-l:t}$ belonging to a probability distribution ϕ is very less then the probability of $X_{n-l:n}$ where $n \leq t-1$. In other words:

$$P(X_{t-l:t}) \in \phi \ll P(X_{n-l:n}) \in \phi \tag{4}$$

   This type of anomaly is called collective anomaly.

3. Probability of point X_t belonging to a probability distribution ϕ is almost same as the probability of previous data points X_n where $n \leq t-1$ but the probability of point X_t belonging to a probability distribution θ is very less than the probability of point X_p belonging to θ. Here In other words:

$$P(X_t) \in \phi \approx P(X_n) \in \phi \tag{5}$$
$$P(X_t) \in \theta \ll P(X_n) \in \theta \tag{6}$$

   Here θ and X_p are probability distribution and data points. *theta* is a distribution defined to a set of local points X_p where $p \leq t$ and p is a set of indices. This type of anomaly is called Contextual anomaly.

Now to detect all the type of anomalies described above it is necessary to estimate the underlying probability distributions responsible for generating the stream. In general there are following few steps in an Anomaly detection framework as described in figure 3.

First step is to pre-process the input data. We use min-max normalization to make sure that all input values are limited between 0 and 1. Min-max normalization can be defined as

$$z = \frac{x - min(x)}{max(x) - min(x)} \tag{7}$$

Second step is to estimate the underlying probability distribution by building a model. This model can be used to model the streaming time series.

Next step would be to score the incoming data point. At this moment we will use the previously estimated probability distribution and types of anomalies defined above to score the incoming data point.

Afterwards the score is passed through a post-processing step. In this step possibly normalization and probabilistic likelihood estimation is applied on top of the previously calculated anomaly score. In other words previously calculated anomaly score is transformed between the range of 0 and 1.

After post-processing, we can use this score to actually rank each incoming data point among previously existing data points. To label each data point as anomaly or not anomaly a thresholding procedure is necessary. Depending upon the type of post-processing extreme cases are considered as anomalies. Thus there are many possible points of innovation in the anomaly detection framework based upon the type of problem we are facing. A quick brief of these likely innovations are following:

1. First possible innovation is done at the pre-processing step. Here it is very helpful to normalize the input data and constrain it between 0 and 1. The benefits of normalization show up while calculating anomaly score as anomaly score will be then close to the range of 0 and 1. Another very important point is

2. Second and a major problem to solve is the algorithm that will estimate the probability distribution of streaming data. This algorithm should have at least following properties:

    - The algorithm should have the capability to differentiate the concept drift with actual anomalies and adapt itself accordingly.
    - The algorithm cannot look ahead in the future values. In other words the algorithm will work on real-time streaming data and will only be dependent upon current and past values.
    - It should be able to train itself ideally, before the arrival of next data point. This condition will make the algorithm fast and online.

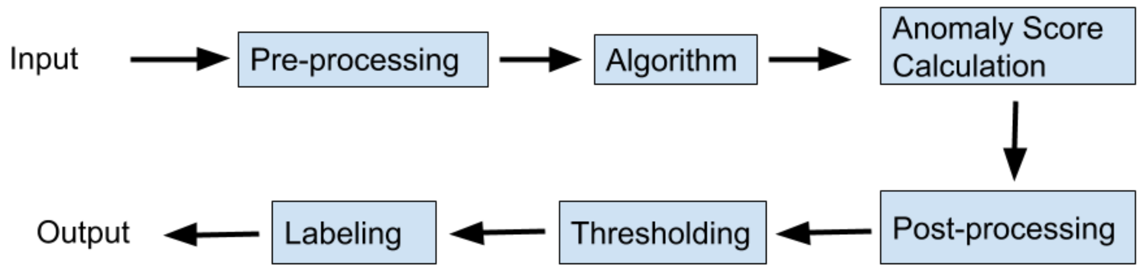


Figure 1: General Anomaly Detection Framework

3. Another possible innovation area is the values the algorithm is trained for. The algorithm can be an autoencoder based and be used to reconstruct a window of current and past values hence predicting a sequence which will then be compared with original values. The distance between the predicted sequence of reconstruction and actual values will be used as anomaly score then. Otherwise using a sub-sequence, one can predict another sub-sequence which is few time step ahead of the input sequence. The autoencoder learns how to reconstruct the current sequence and predictive learns how to predict the future values.

4. Now that we have an anomaly score, next step is to apply some post-processing on top of this score. There are two possible variations used. First variation is moving min-max normalization. With this kind of normalization current point is normalized by using the minimum and maximum values of the data obtained so far. Seoncd type of post-processing is probability of values of a current small window from the previous values.

2 Related Work and State of the art 10 pages

1. Time series intro. Prediction and classification.
2. Unsupervised methods on time series
3. Anomaly detection
4. Time series Anomaly detection
5. Unsupervised Time series Anomaly detection
6. General components latest anomaly detection methods and explanation of each
7. HTM based anomaly detection paper review. Problems or way it does it
8. paper 2
9. paper 3
10. general review, problems, bottlenecks and trend

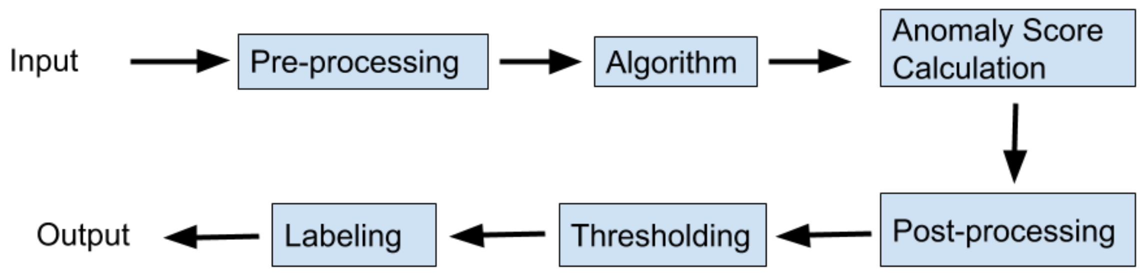


Figure 2: General Anomaly Detection Framework

3 Proposed method 10 pages

3.1 General anomaly score based architecture and detailed explanation

Considering figure 3, each algorithm will be using all parts with few variations and a cusotmized algorithm part.

Pre-processing is playing crucial role in the generalization of the algorithms. There are two different kind of normalization techniques tested. First kind is global normalization Proposed method can be divided in multiple parts. First part would be the implementation of baselines using latest deep learning architectural components. We use these components to develop two different kind of detectors.

One detector is based upon prediction of a next data point. Let $T_{(t-L-1):(t-1)}$ is the small window taken from time series. The starting point is $t - L - 1$ where L is the window size and last data point is taken from $t - 1$ which is the data point at previous time step. This data window is used to predict the time series value at time t. To predict at time t various different kind of functions as baselines using latest architectural components are designed. Generally the output of such a function would be

$$f(T_{(t-L-1):(t-1)}) = \hat{f}(t) \tag{8}$$

Thus the function f in prediction based models map data points as: $f : L \mapsto 1$ where L is the window size and output is the prediction of value at current time. For optimization of function f, the difference between time series value at current time step and predicted value at current time is taken. More specifically the objective function will be

$$min|\hat{f}(t) - f(t)| \tag{9}$$

That is to minimize the difference between time series value at time t and predicted time series value at time t using time series value from $T_{(t-L-1):(t-1)}$.

3.2 Describe autoencoder based architecture

An autoencoder is a network which is trained to copy its input to its output, with the typical purpose of dimension reduction - the process of reducing the number of random variables under consideration. It features an encoder function to create a hidden layer (or multiple layers) which contains a code to describe the input. There is then a decoder which creates a reconstruction of the input from the hidden layer. An autoencoder can then become useful by having a hidden layer smaller than the input layer, forcing it to create a compressed representation of the data in the hidden layer by learning correlations in the data. This compressed representation is actually summarized version of input. This way for the anomaly input values, summarized version will not output very similar output and hence ending up creating big reconstruction loss. The backpropagation will try to $h_{W,b}(x) \approx x$, so as to minimise the mean square difference:

$$L(x, y) = \sum (x - h_{W,b}(x))^2 \tag{10}$$

Similarly for time series let X_t be the input value of the streaming data at time t then the input of autoencoder based architecture will be $X_{t-w:t}$ and output will be $Y_{t-w:t}$ as Y is the predicted reconstruction of the input.

$$Y_{t-w:t} = h(X_{t-w:t}) \tag{11}$$

The reconstruction loss is defined as:

$$L(x, y) = \frac{1}{w} \sum (Y_{t-w:t} - X_{t-w:t})^2 \tag{12}$$

$\gg\ll$ picture of an autoencoder

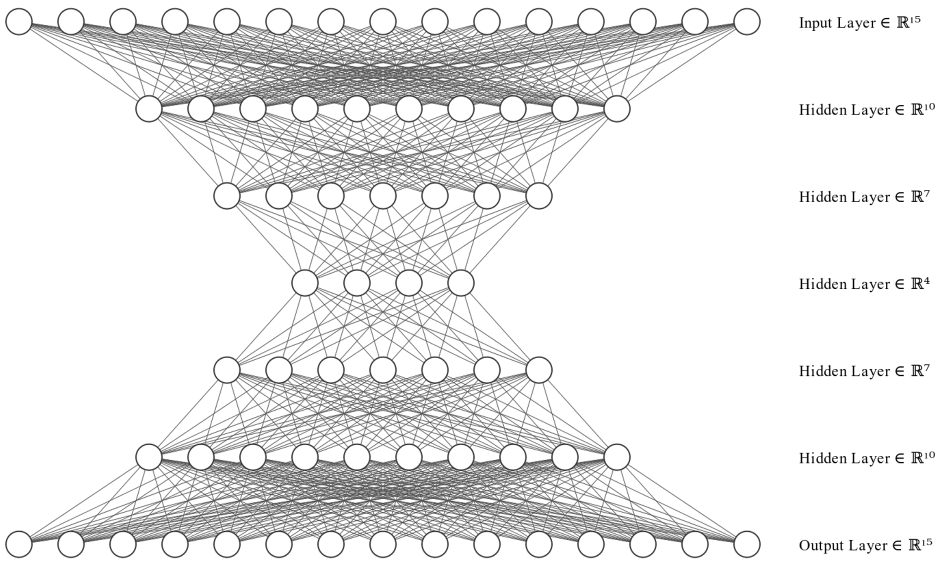


Figure 3: Fully connected autoencoder

3.3 Describe Multistep-Ahead based architecture

$\gg\ll$ picture of an multi-step based architecture Second type of architecture used is multi-step prediction architecture. In this type of architecutre For time series let X_t be the input value of the streaming data at time t then the input of multistep prediction based architecture will be $X_{t-w-l:t-l}$ and output will be $Y_{t-l:t}$ as Y is the prediction of the input. More specifically

$$Y_{t-l:t} = h(X_{t-w-l:t-l}) \tag{13}$$

and the prediction loss will be defined as:

$$L(x, y) = \frac{1}{l} \sum (Y_{t-l:t} - h(X_{t-w-l:t-l}))^2 \tag{14}$$

Above two are the basic outputs which are repeated with following different architectures.

3.4 Describe fully connected architecture

3.5 Describe fully connected prediction architecture

3.6 Describe fully connected autoencoder architecture

3.7 Describe Convolution based prediction architecture

3.8 Describe LSTM based prediction architecture

3.9 Describe LSTM based autoencoder based architecture

3.10 Recency concept

3.11 Post-processing

3.12 Thresholding

3.13 Parameterinzing the anomaly score

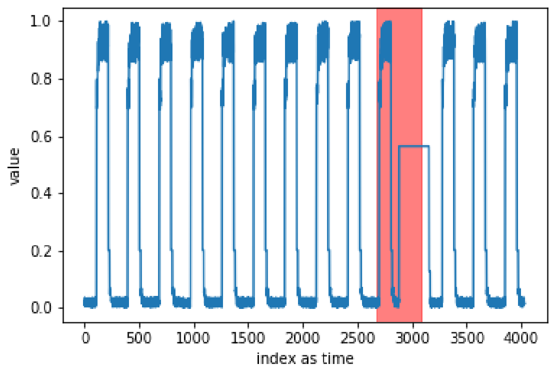
4 Empirical Formulation and Experiments 5 pages

To experiment with proposed method and campare them with state-of-the-art results Numenta Anomaly Benchmark (NAB) was used. Numenta is a benchmark specifically designed for online time series anomaly detection.

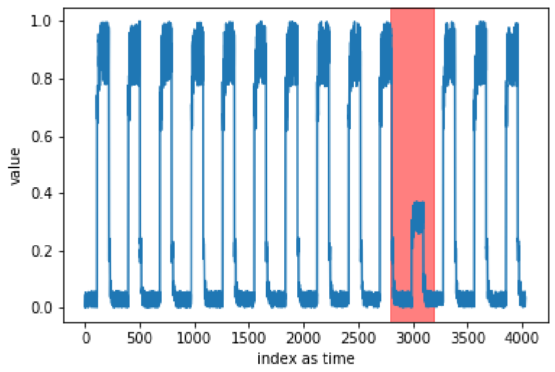
Using NAB provides with many benefits. First is a benchmark dataset which 58 various univariate time series, each with timestamp as index and value. Length of each time series ranges from 1000 to 22000 data points. These 58 time series are from six different categories. These categories include artificially generated time series, times series gathered from advertisement industry related data such as click through rate (CTR), AWS usage metrics, server traffic data, tweets volume time series and known cause anomaly which has the a solid reason for the labeled anomaly.

The diversity of these categories provide the second benefit. With the diversity almost all type of anomalies are covered hence each algorithm can be tested against each type. Artificial datasets further contains time series with anomalies and without anomalies. Datasets without anomalies contain only normal patterns and no anomaly thus any predicted anomaly will be a false positive here.

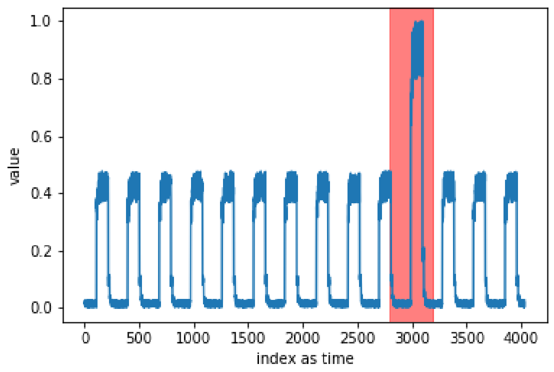
4.1 Dataset



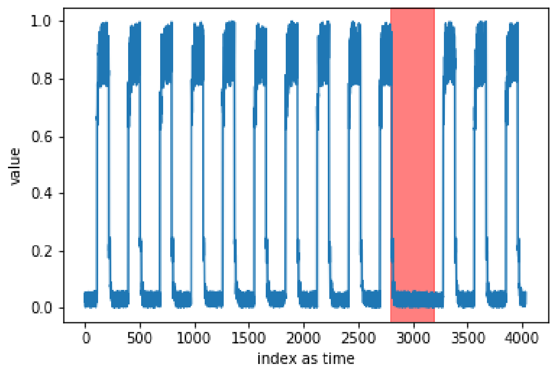
(a) Artificial with sudden flat-middle



(b) Artificial with sudden jumps-down



(c) Artificial with sudden jumps-up



(d) Artificial with sudden flat

Figure 4: Examples of artifical anomaly data set

5 Results 10 pages

5.1 Scoring

5.2 Comparing with others

6 Conclusion and Discussion 5 pages

7 Experiment Infrastructure

7.1 Experiment Management using MLflow

7.2 Parallel execution using Docker

8 Best practices

Following steps were taken to maximize the efficiency and speed of research:

1. Use version control to track the code and share between different devices.
2. Separate code from data. This will keep the code base small and easy to debug.
3. Separate input data,working data and output data.
    - **Input Data:** Input data-set that never change. For my case it is NAB and other external datasets.
    - **Working Data:** nothing for now.
    - **Output Data:** Results and threshold profiles in my case.
4. Separate options from parameter. This is important:
    - Options specify how your algorithm should run. For example data path, working directory and result directory path, epochs, learning rate and so on.
    - parameters are the result of training data. it includes the score and hyper-parameters.

8.1 Moving from jupyterlab to pycharm

While working with jupyterlab notebook following routine was followed: 1- Load data with sample function 2- Write an algorithm 3- Test the results 4- Write general executeable .py file. 5- Get results on server

Since we needed to track change on two different places, it was becoming harder to track the bugs and improve on efficiency. That's why pycharm was selected to create executeable files and test algorithms at the same time.

9 Reference Usage

10 References

‘