Unsupervised anomaly detection

in multivariate time series data

Damian Horna

November 8, 2019

Motivation - application domains

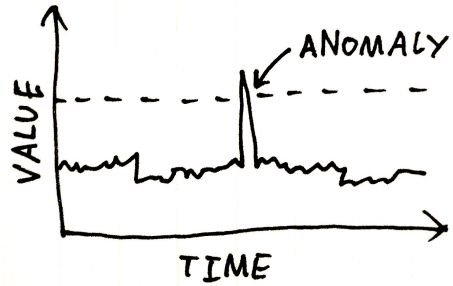
The importance of anomaly detection is due to the fact that anomalies can occur in many domains, e.g.:

- Fraud detection
- Manufacturing
- Cybersecurity
- Medicine

What is an anomaly?

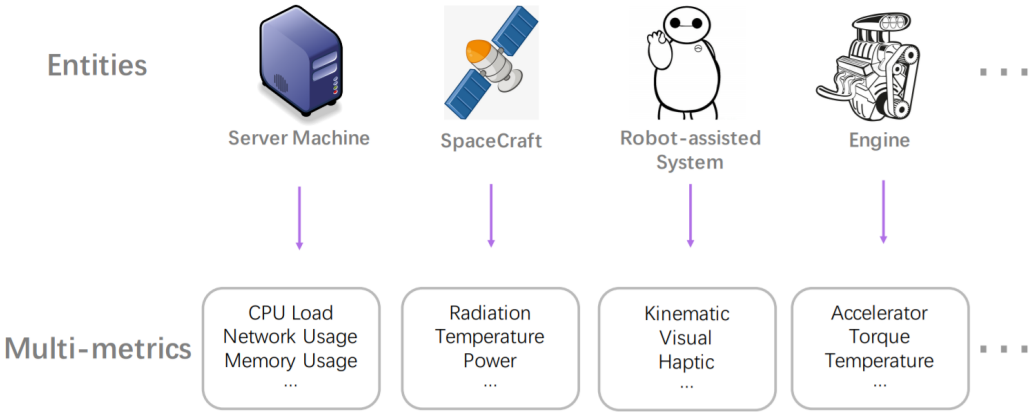
"An anomaly is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism."

D. M. Hawkins



Visualization of an anomaly Source

Entities with monitored multivariate time series



Entities with monitored multivariate time series Source

Multivariate time series - an example

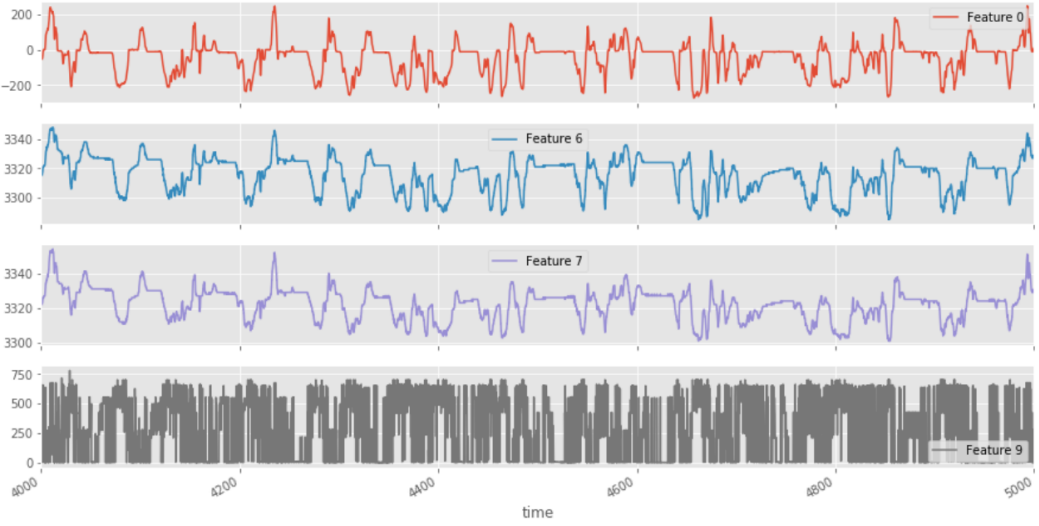


Figure: Continuous signals

Multivariate time series - another example

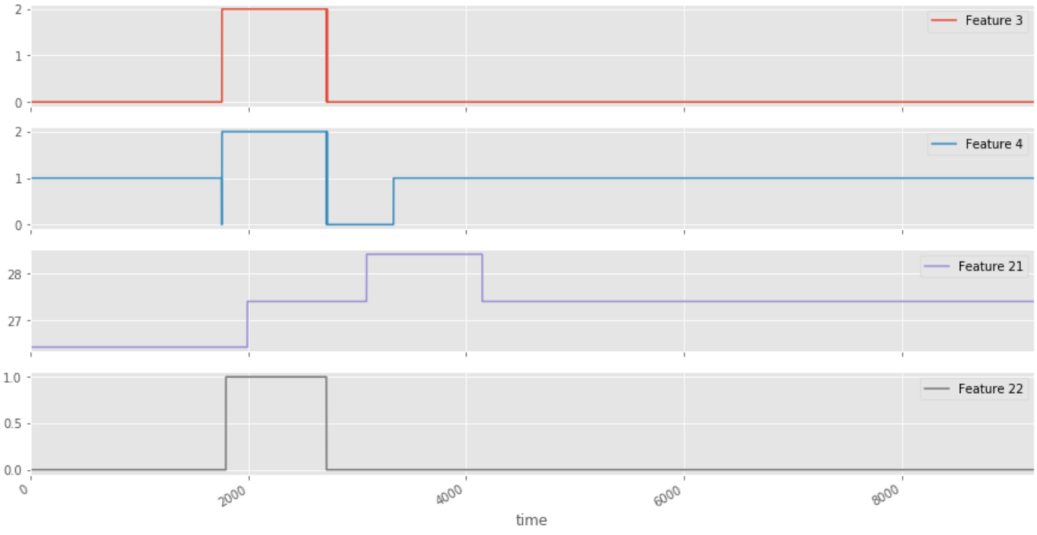


Figure: Discrete sequences

Supervised anomaly detection

When the labels are available, anomaly detection basically is a special case of the classification problem. It may not be the easiest classification problem to solve after all, due to problematic characteristics of anomalous data, i.e.:

- Class imbalance - regarding the definition of an outlier, data distribution is very skewed.
- Contaminated normal class examples - some of the data instances which are labeled as "normal" may not be so normal after all.
- Small sample size - often there are only few anomalies that we are sure of, but there may be more anomalies that just aren't present in the data.

Unsupervised anomaly detection methods

Unsupervised anomaly detection is even more challenging because we need to find the answer ourselves to what is normal and what anomalous.

There are several approaches one can take:

- Adjust some of the supervised methods for anomaly detection (e.g. Isolation forest)
- Clustering algorithms: DBSCAN, K-means etc.
- Other: HBOS, one-class SVM, LOF

Nevertheless all promising methods assume that we feed them with mostly "normal" data, so we need to at least understand what is "normal".

So how do we go about anomaly detection in time series?

Time series data is different, because of the time aspect and almost all previously discussed methods completely ignore the temporal dependencies of the data. Therefore some of them may be useful in detection of point anomalies, but on the whole they tend to perform poorly.

When detecting anomalies in time series data we can choose one of the two strategies:

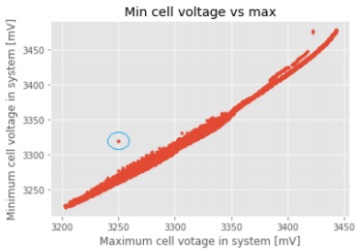
- Prediction-based approach
- Reconstruction-based approach

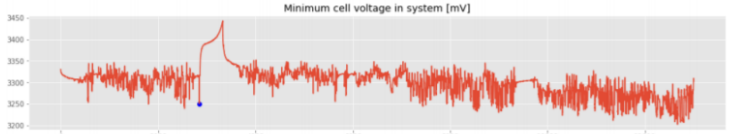
And sometimes we can even mix them together. However, both approaches require that we use mostly "normal" data for training.

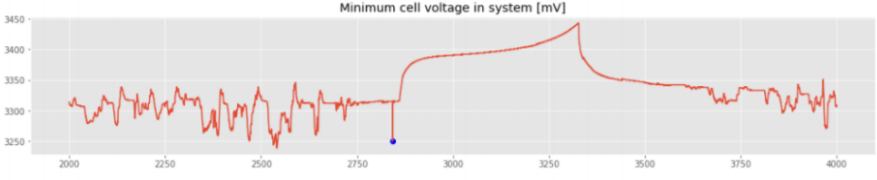
Our approach

1. Exploratory data analysis and feature engineering
2. Generation of synthetic anomalies
3. Test-bed construction
4. Evaluation of detection algorithms

Outlier identification

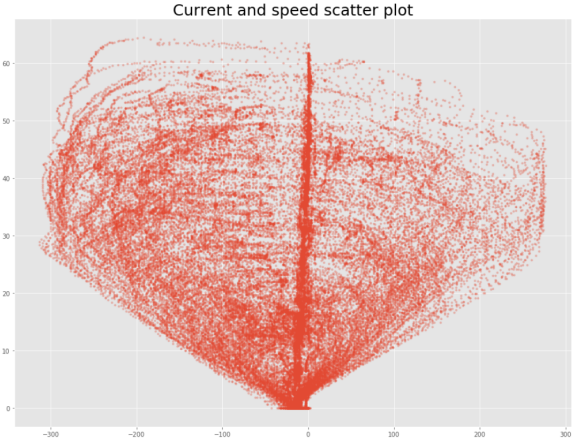






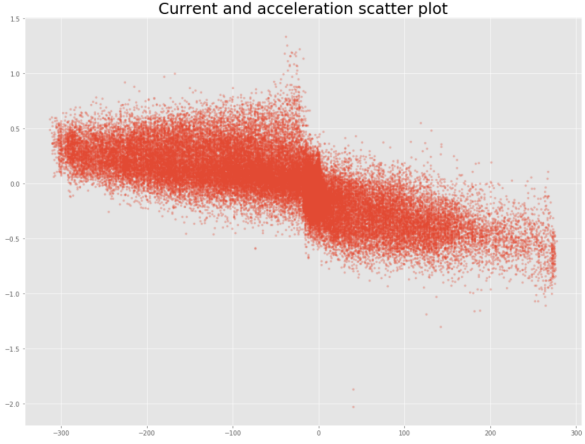
Inter-correlations

We examine inter-correlations between all pairs of time series, but in many cases there is no sign of correlation.



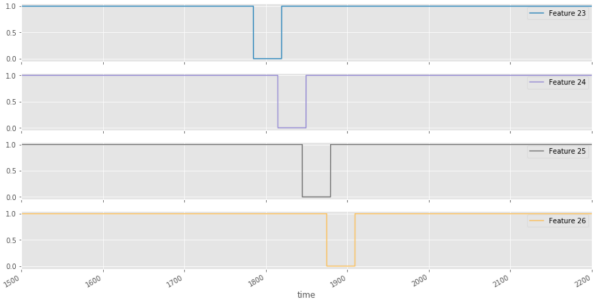
Feature engineering

However, sometimes feature engineering is of some help. In this case we used speed to derive acceleration and found strong negative correlation:



Feature engineering - more ideas

- Lag features

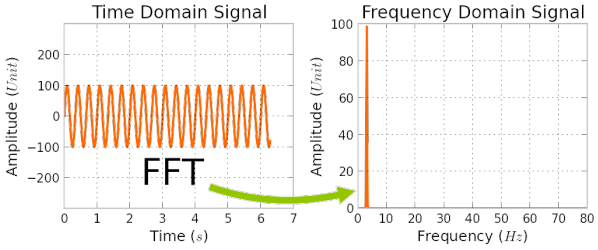


Feature engineering - more ideas

- Lag features
- Mean/median over subsequences

Feature engineering - more ideas

- Lag features
- Mean/median over subsequences
- Discrete Fourier Transform, Discrete Wavelet Transform



FFT visualization Source

Feature engineering - more ideas

- Lag features
- Mean/median over subsequences
- Discrete Fourier Transform, Discrete Wavelet Transform
- Differentiation

Feature engineering - more ideas

- Lag features
- Mean/median over subsequences
- Discrete Fourier Transform, Discrete Wavelet Transform
- Differentiation

And more standard:

- Logarithm
- Polynomials of features
- Square root

Imperfections of sensor data

Sensor data is often associated with certain problems, namely:

- Irregular time intervals between sensor readings
- Sensor noise

These imperfections may be a sign of different type of anomalies themselves.

Time-gaps and noise example

Suspicious speed sensor readings:

49527.0	0.000000
49527.1	0.000000
49527.2	0.000000
49527.3	0.000000
49527.4	0.000000
49527.5	0.000000
49576.3	0.000000
49576.4	50.984375
49576.5	51.070312
49576.6	51.242188
49576.7	50.945312
49576.8	50.863281
49576.9	50.910156
49577.0	51.121094

23138.0	0.000000
23138.1	0.000000
23138.2	0.000000
23138.3	0.000000
23138.4	0.906250
23138.5	0.906250
23138.6	149.648438
23138.7	2.765625
23138.8	2.277344
23138.9	2.800781
23139.0	3.378906
23139.1	3.886719
23139.2	4.261719

Noise reduction

time	Feature 0	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7
0.016293	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.017833	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.020790	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.022277	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.027354	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.053726	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.065602	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.066875	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.068793	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.070502	-209.0	1.0	0.0	0.0	1.0	30.687500	3265.0	3300.0
0.115263	-206.0	1.0	0.0	0.0	1.0	30.687500	3265.0	3300.0
0.117847	-206.0	1.0	0.0	0.0	1.0	30.687500	3265.0	3300.0

```
df.groupby(round(df['time'], self.precision)).median()
```

Compression

Result:

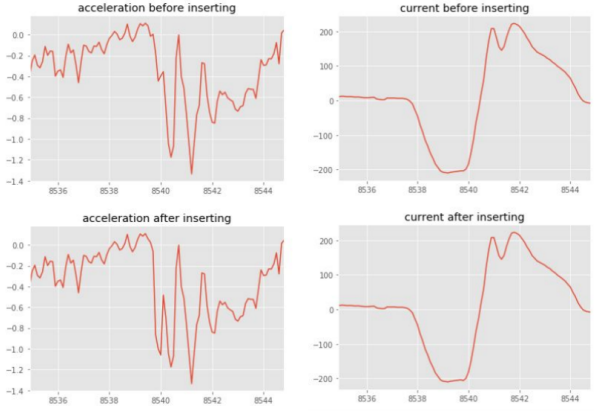
time	Feature 0	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7
0.0	-209.0	1.0	0.0	0.0	1.0	30.417969	3265.0	3300.0
0.1	-206.0	1.0	0.0	0.0	1.0	30.687500	3265.0	3300.0

And if you want to compress even more:

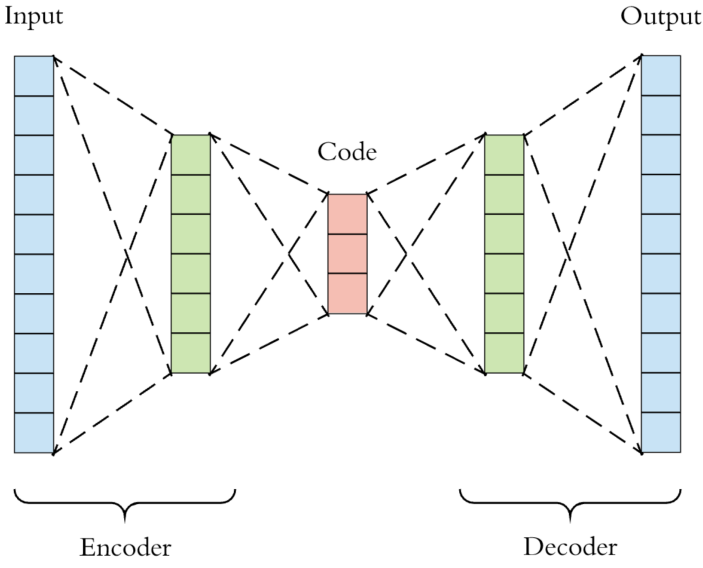
```
df.astype(np.float32)
```

Test-bed construction

In order to evaluate detection algorithms qualitatively, there is a need for test-bed containing *normal* and anomalous examples as well. We divide the data we have into *normal* and anomalous. Furthermore we also generate artificial anomalous examples to compare the algorithms on more data.

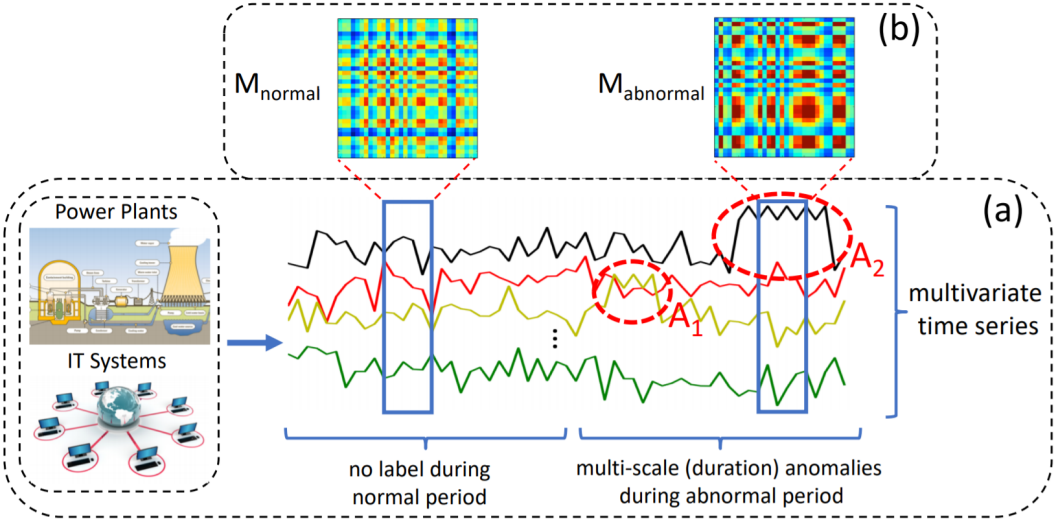


Reconstruction-based detection: autoencoder



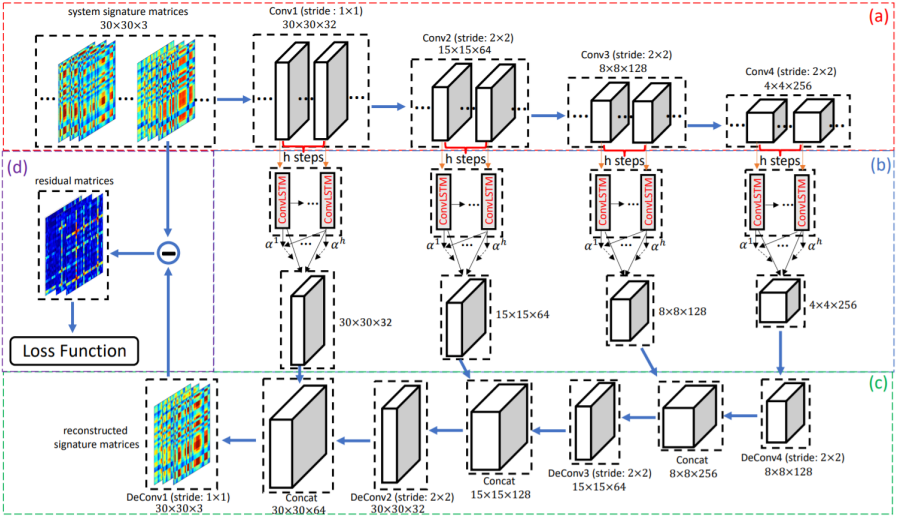
Autoencoder architecture Source

MSCRED



MSCRED Source

MSCRED architecture



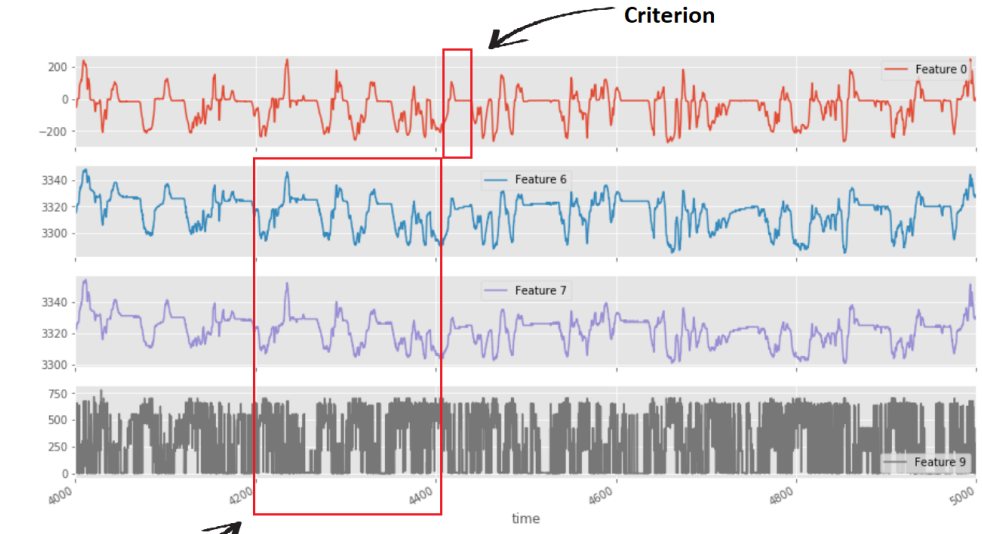
MSCRED architecture Source

Prediction-based anomaly detection: LASSO

LASSO regression was applied as a prediction-based method for contextual anomaly detection using data from multiple time series. One of the biggest advantages of LASSO regression is that it automatically performs feature selection thanks to its loss function:

$$\sum_{i=1}^{n}(y_i - \sum_{j} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

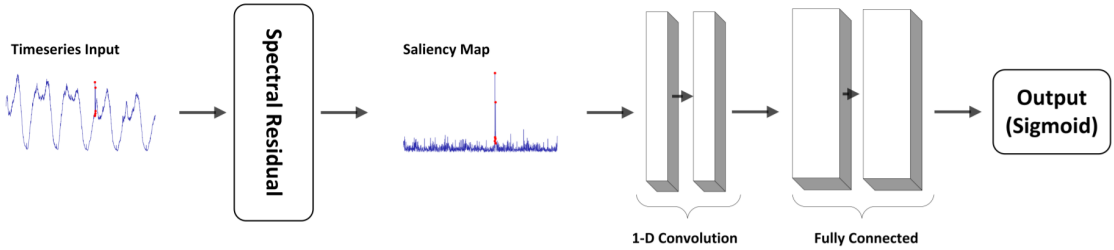
LASSO prediction



LASSO prediction example

LASSO prediction example:

SR-CNN - an overview



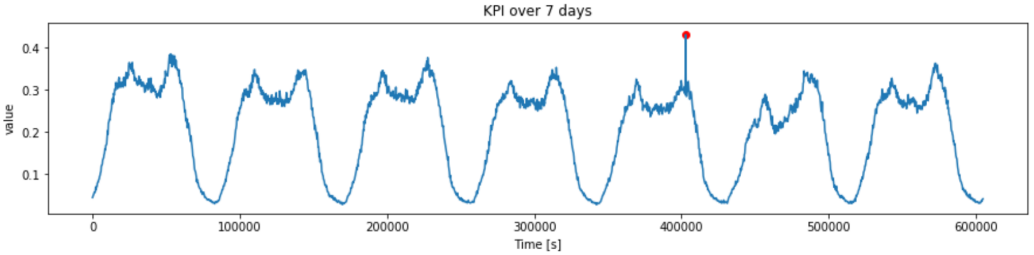
SR-CNN architecture Source

Spectral Residual overview

$$A(f) = Amplitude(\mathfrak{F}(\mathbf{x}))$$
$$P(f) = Phrase(\mathfrak{F}(\mathbf{x}))$$
$$L(f) = log(A(f))$$
$$AL(f) = h_q(f) \cdot L(f)$$
$$R(f) = L(f) - AL(f)$$
$$S(\mathbf{x}) = \left\| \mathfrak{F}^{-1}(exp(R(f) + iP(f))) \right\|$$

Saliency map calculation Source

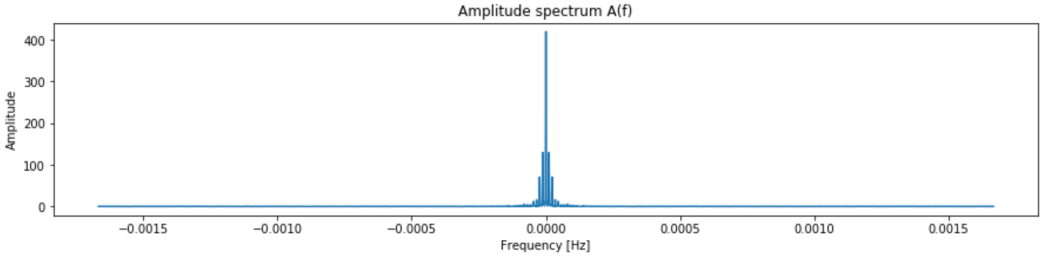
SR step by step: time series



Spectral residual Source

SR step by step: amplitude spectrum

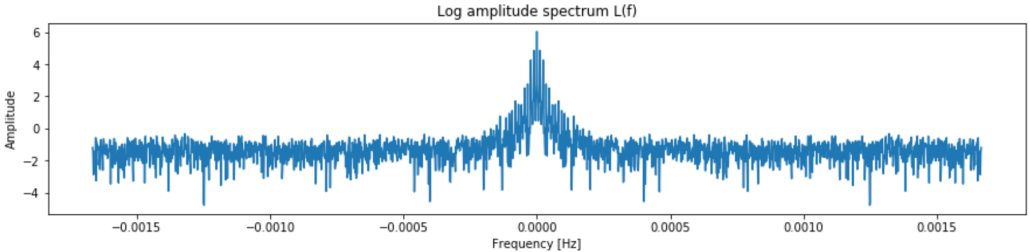
$$A(f) = Amplitude(\mathfrak{F}(\mathbf{x}))$$



Spectral residual Source

SR step by step: log amplitude spectrum

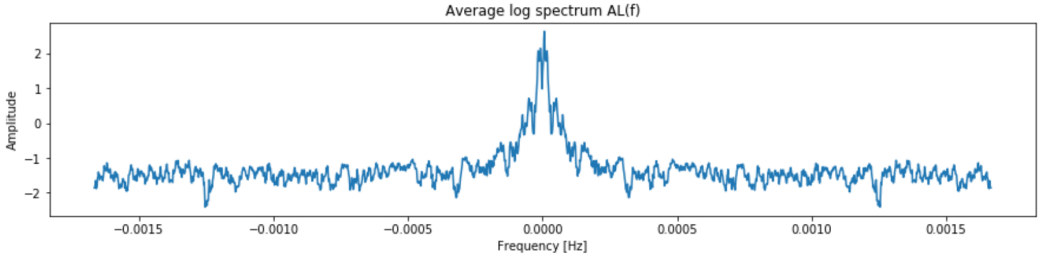
$$L(f) = log(A(f))$$



Spectral residual Source

SR step by step: average log amplitude spectrum

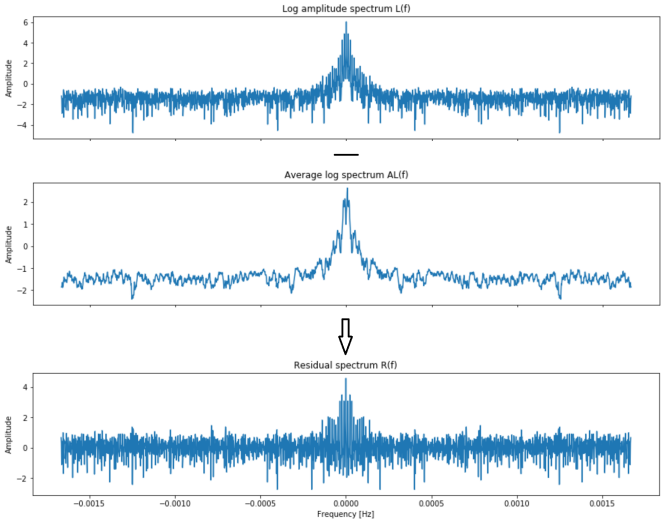
$$AL(f) = h_q(f) \cdot L(f)$$



Spectral residual Source

SR step by step: residual spectrum

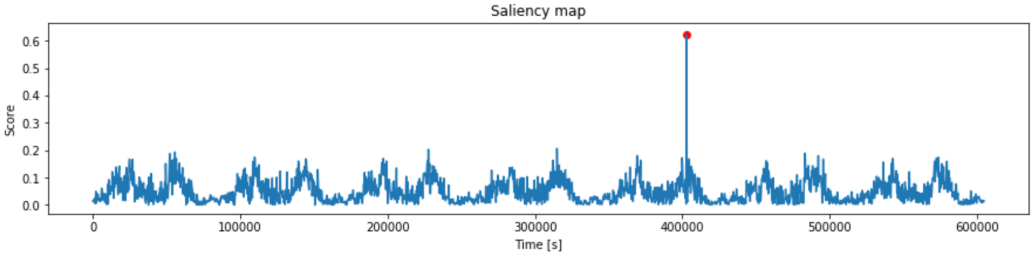
$$R(f) = L(f) - AL(f)$$



Spectral residual Source

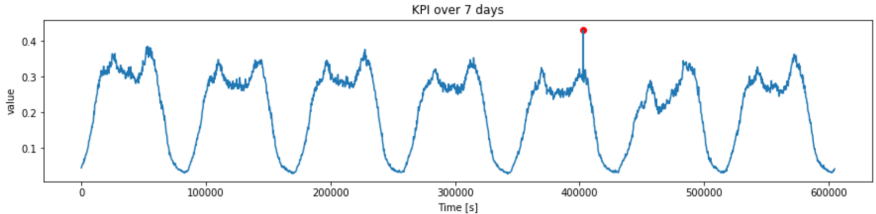
SR step by step: inverse FFT

$$S(\mathbf{x}) = \left\| \mathfrak{F}^{-1}(exp(R(f) + iP(f))) \right\|$$



Spectral residual Source

SR step by step: before and after



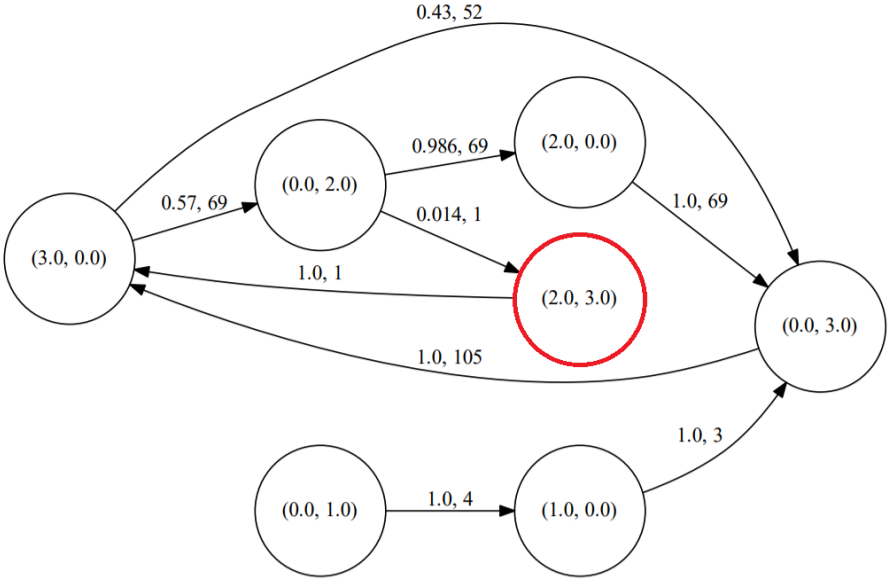
Spectral residual Source

Markovian model - anomaly detection in discrete sequences

Essentially it's a special kind of *Finite State Automaton*.

1. Each state is represented by a sequence of symbols a_i from the alphabet Σ
2. Transition corresponds to an event, representing addition of new symbol at the end of the sequence.
3. Each transition is therefore associated with the probability $P(a_n|a_{n-k}...a_{n-1})$
4. Parameter k represents the *memory* of the model.

Markovian model - an example



Summary

We presented some practical problems that can occur when dealing with multivariate time series data and how to solve them, as well as state of the art algorithms for unsupervised anomaly detection. However, there are plenty of methods that we did not talk about, e.g. LSTM recurrent neural nets. We highly encourage to dig in a little more into the subject, as it has a lot of practical use cases regarding The Fourth Industrial Revolution that we are witnessing right now.

References

- Charu C. Aggarwal. “Outlier Analysis Second Edition”. Springer, 2017.
- Time-Series Anomaly Detection Service at Microsoft
- A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data
- Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network
- A Comparative Evaluation of Anomaly Detection Techniques on Multivariate Time Series Data

Group of
Horribly
Optimistic
STatisticians