

Marvin Arnold  
Faculty of Computer Science

# Unsupervised Machine Learning for Anomaly Detection using an Autoencoder

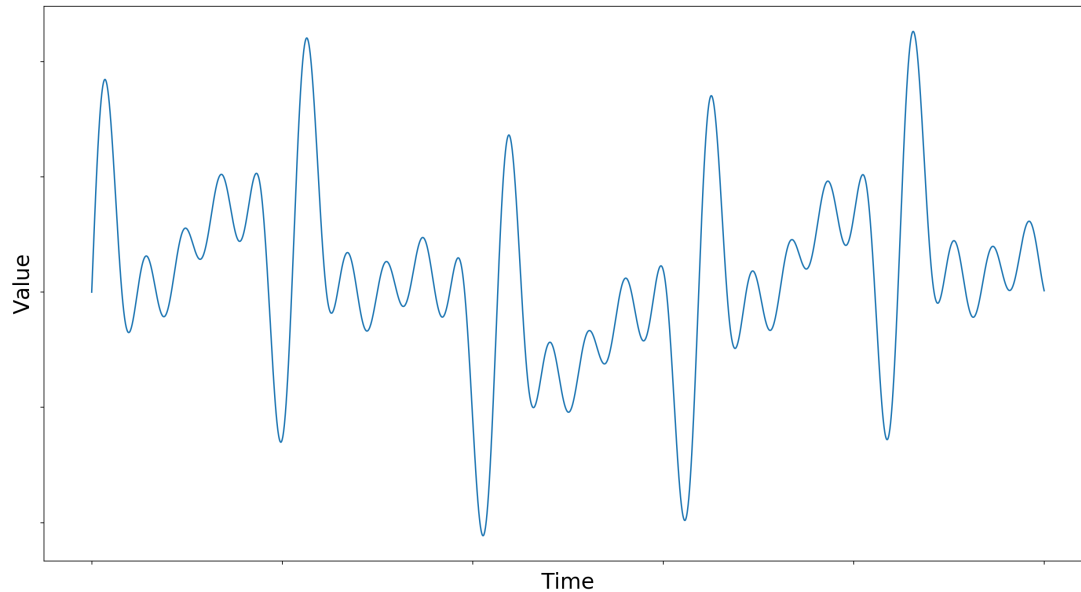
Profilprojekt Anwendungsforschung in der Informatik  
xx.03.2020

# Agenda

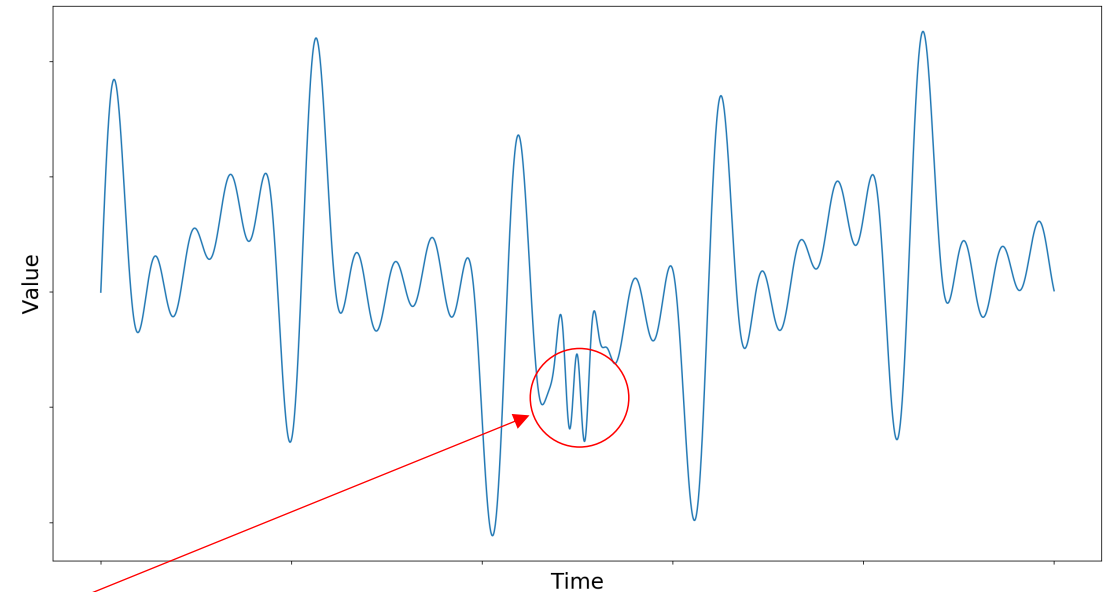
# Motivation - Anomaly Detection

- Detect abnormal behaviour/patterns of measurements of a machine
- **Catch:** It is unknown how the anomaly manifests itself in the measurement data (think nuclear power plant)

Normal behaviour



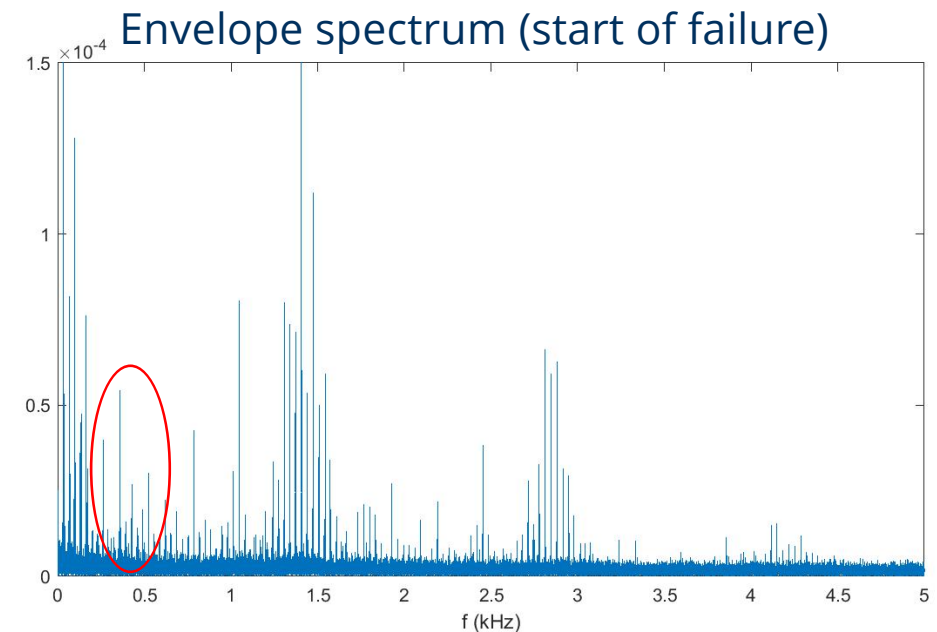
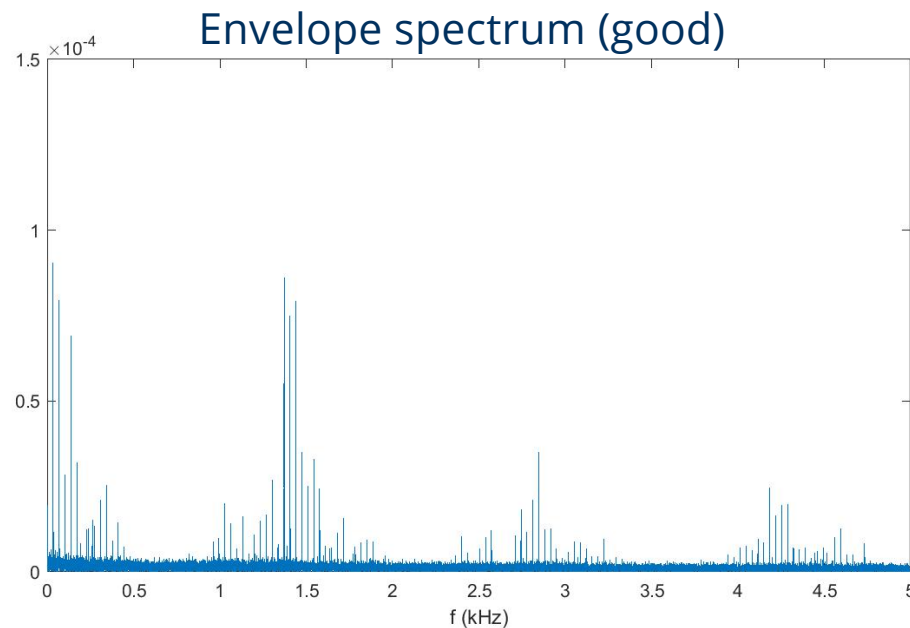
Possible error behaviour



Previously unknown behaviour. Is this an error or just noise/measurement error?

# Available Data

- Bearings running until they break → 2 Datasets available ("**Lager 4**" and "**Lager 5**")
- Every 2 minutes 4 separate measurements from 4 ultrasonic sensors
- Available "ground truth" data: Raw signal, amplitude spectrum, spectrogram, **envelope spectrum**

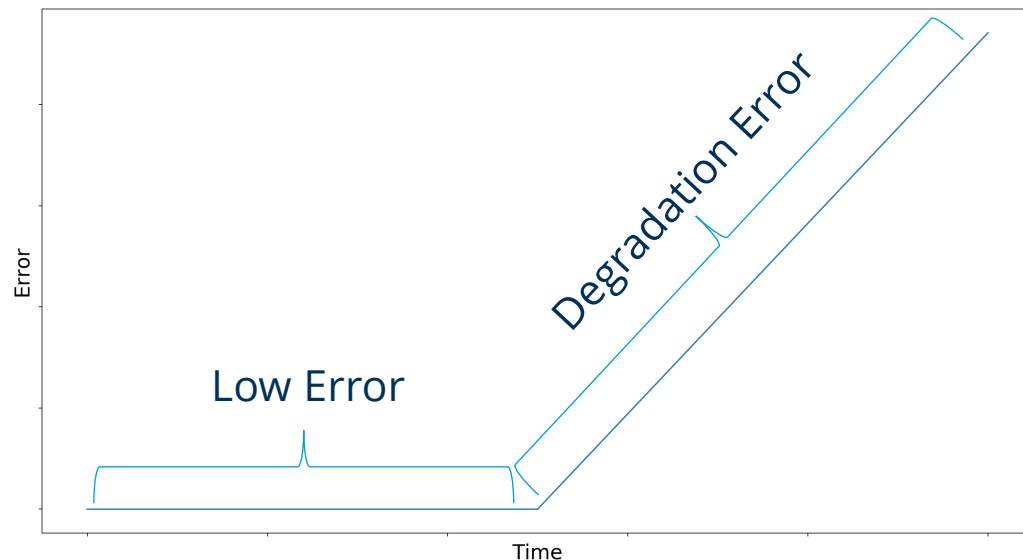


- **Expert Feature:** Specific area in envelope spectrum that shows more noise as bearing is breaking
- Knowledge of expert feature is very problem specific

# Task

Raw audio bearing should contain all the necessary information to detect the starting point of the degradation.

Use an appropriate machine learning scheme to map non pre-processed audio recording to an error plot.



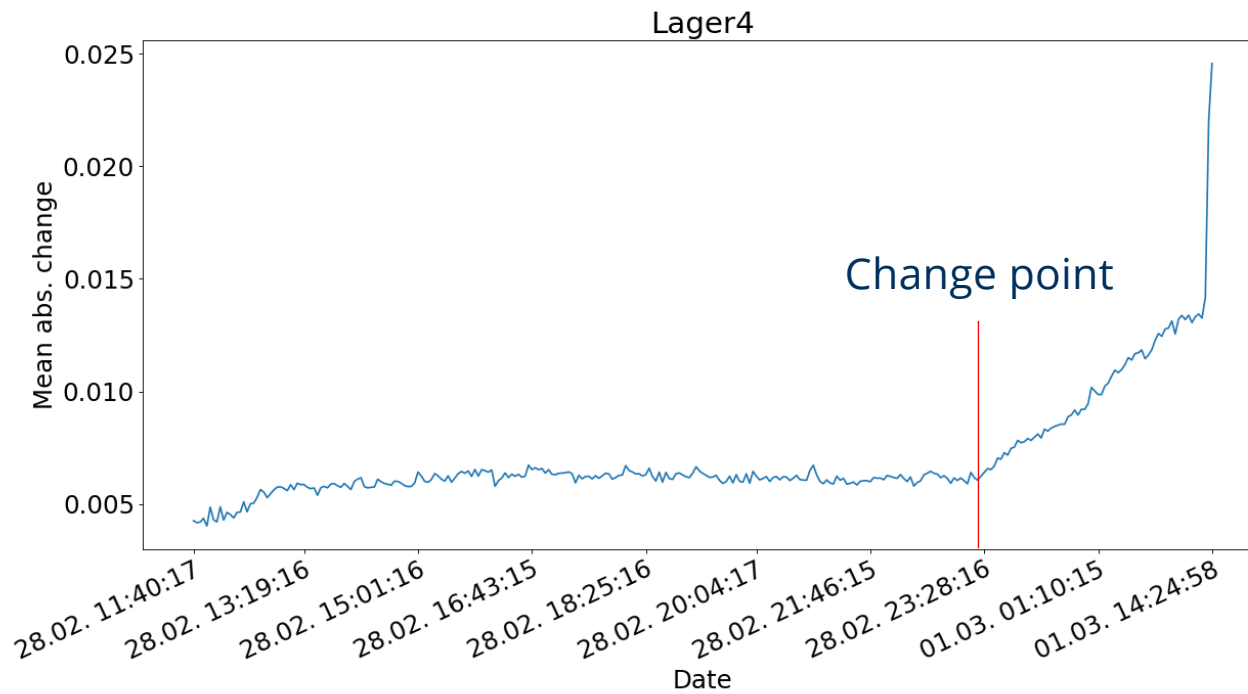
## Main Idea:

- Using an error plot the start of degradation should be easily visible
- Reduction of expert feature possibilities to universal error plot
- Evaluation using ROC and AUC

# Classical approaches (non machine learning)

- We calculate 65 classical statistical features and search for a separability threshold
- This includes: autocorrelation, kurtosis, linear trend, mean, median, skewness, variance, ...

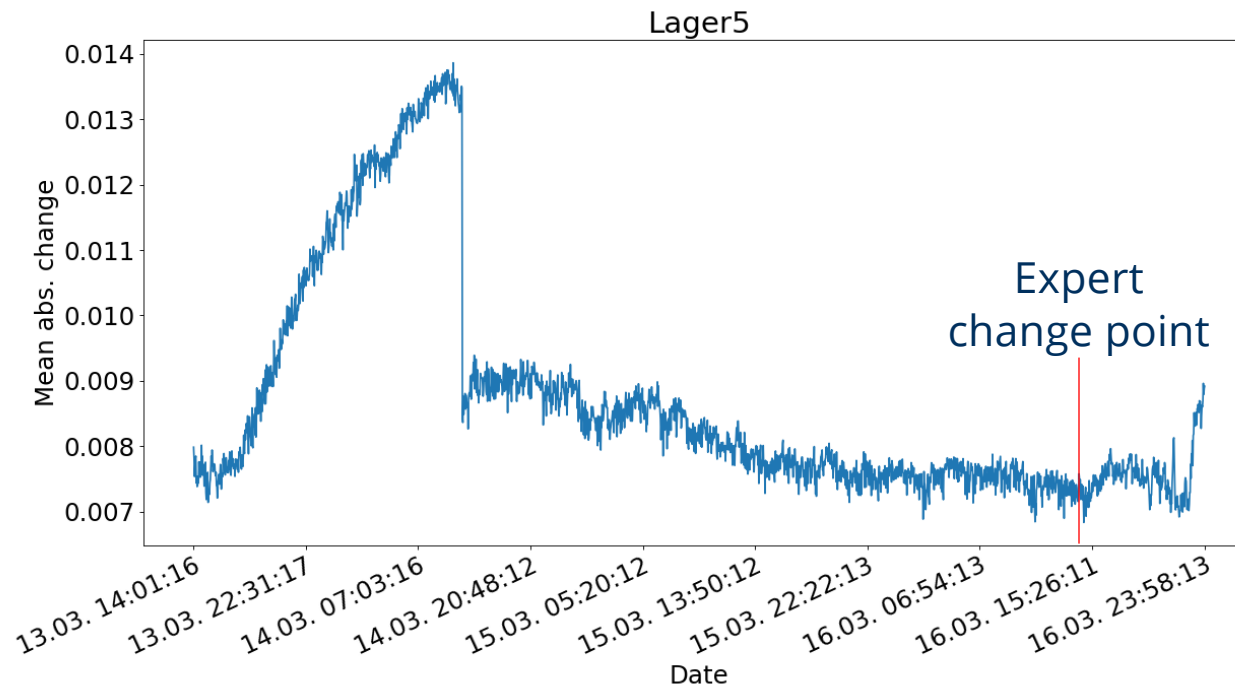
## 1. Mean absolute change:



- Describes the absolute change between two consecutive measurement points
- Error = Mean abs. change
- Error curve looks nearly ideal
- Change point almost perfectly hits the expert opinion
- AUC = 0.99

# Problems with mean absolute change

- **Lager 5** does not look so nice ...

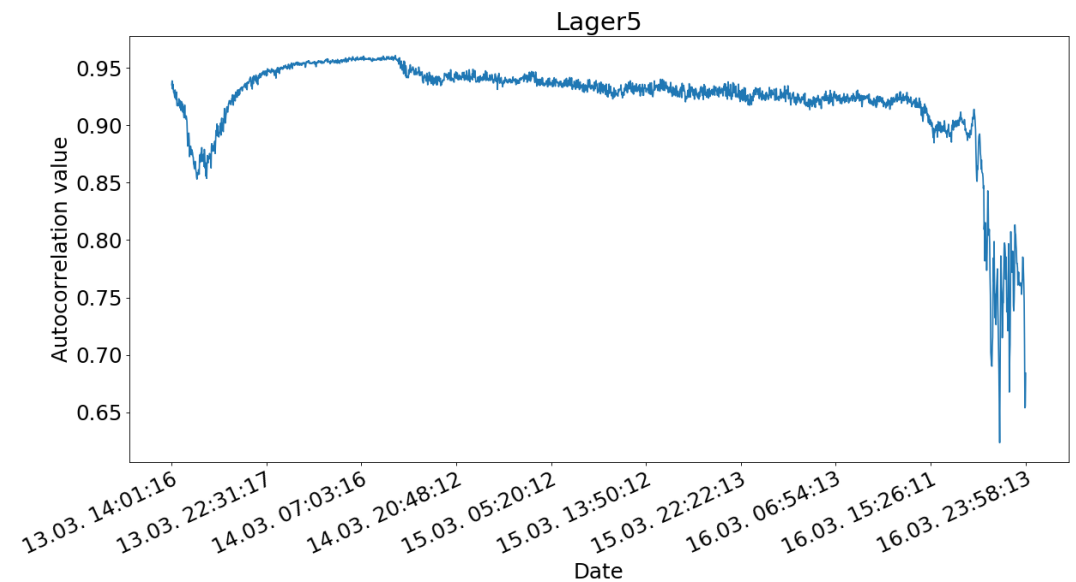
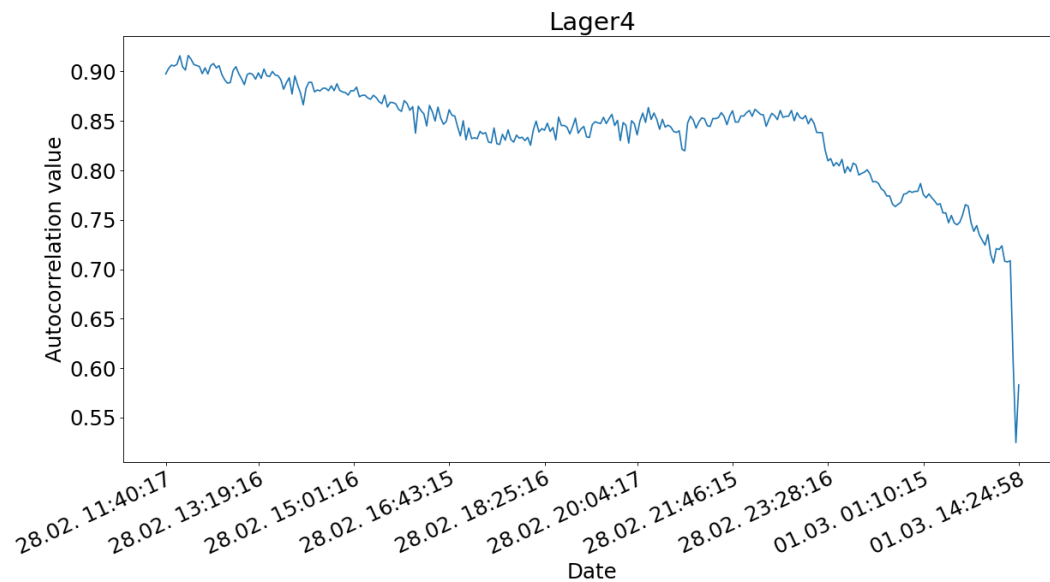


- Error seems to happen right at the start
- Afterwards big drop in error
- Error after change point is not increasing
- AUC = 0.21 (classifier could be reversed)

→ Does not work universally

# Autocorrelation

- As bearing rotates one full revolution measurement values should correlate with previous revolution
- Using the microphones sampling frequency → 6144 Measurement samples ↔ one revolution



- **Lager 4** AUC = 1.00
- **Lager 5** AUC = 0.98



# Problems with autocorrelation

- We need to specify the autocorrelation lag → we know the exact lag of 6144 thus the results are very good
- What if we don't know the “correct” lag?

**AUC values for different lags**

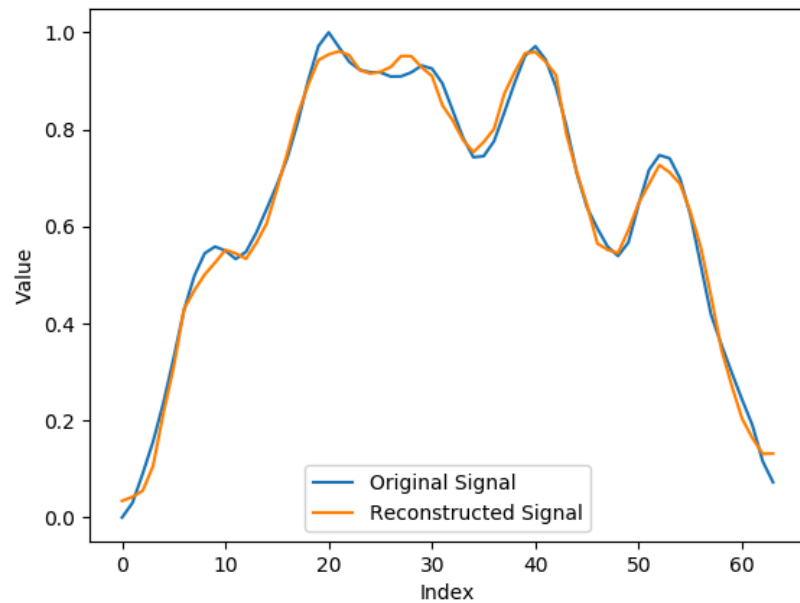
	128	256	512	1024	2048	4096	8192	16384
Lager 4	0.94	0.78	0.81	0.84	0.59	0.76	0.57	0.60
Lager 5	0.98	0.89	0.95	0.69	0.93	0.92	0.94	0.86

- Guessing the lag gets us random good and bad results
- We require knowledge of the underlying problem to find optimal lag value

# Machine Learning Approach - Autoencoder

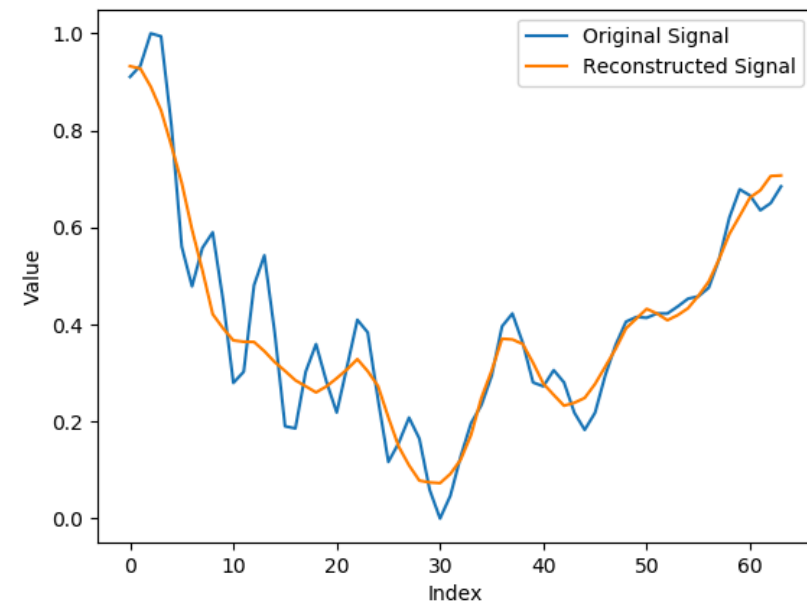
- Reconstruction of the  $[0, 1]$  normalized original signal
- We expect the reconstruction of a healthy signal to be better than on a signal showing degradation
- **Evaluation metric:** Sum of the quadratic error of each time step in each measurement

## Good reconstruction

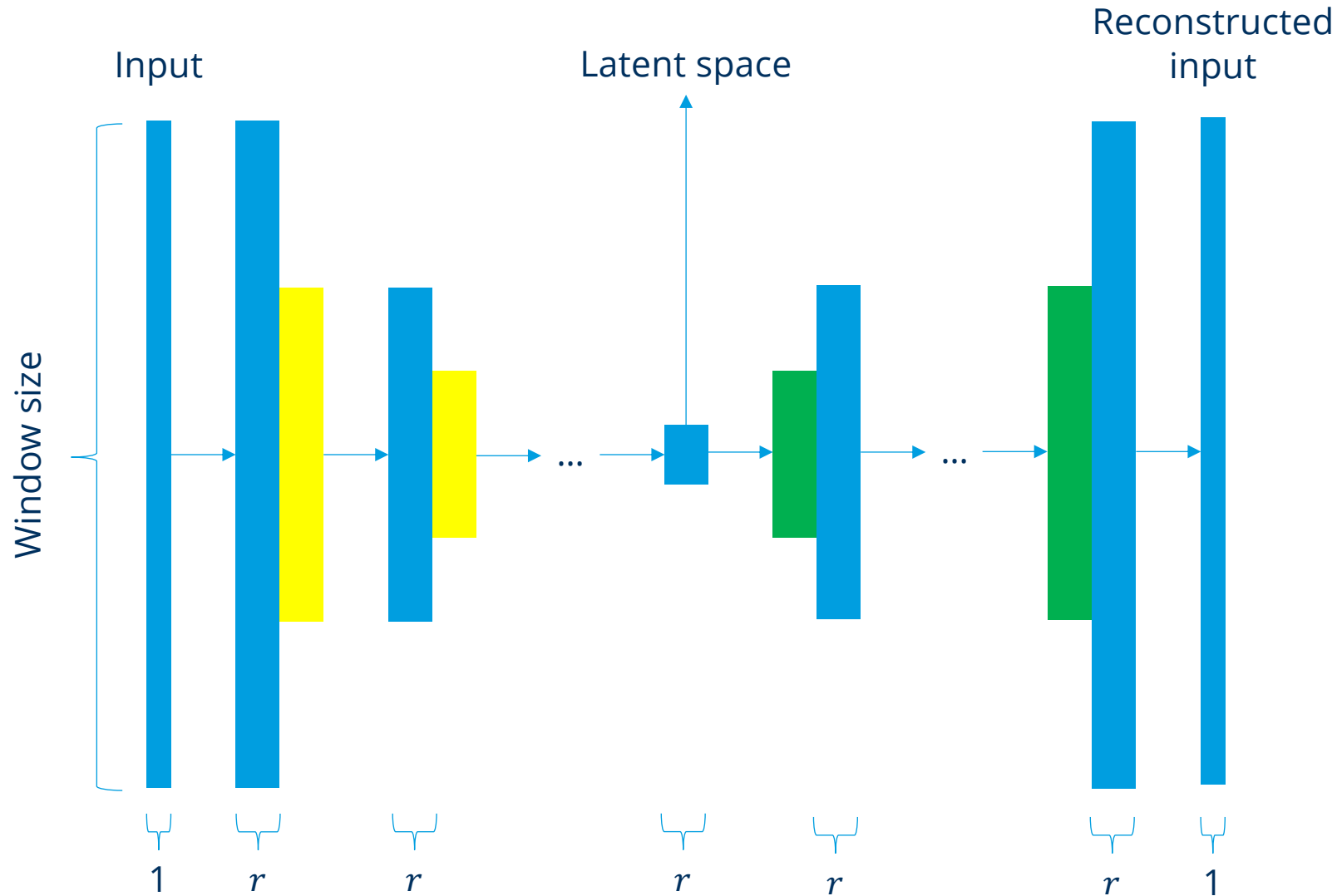


Degradation  
happens →

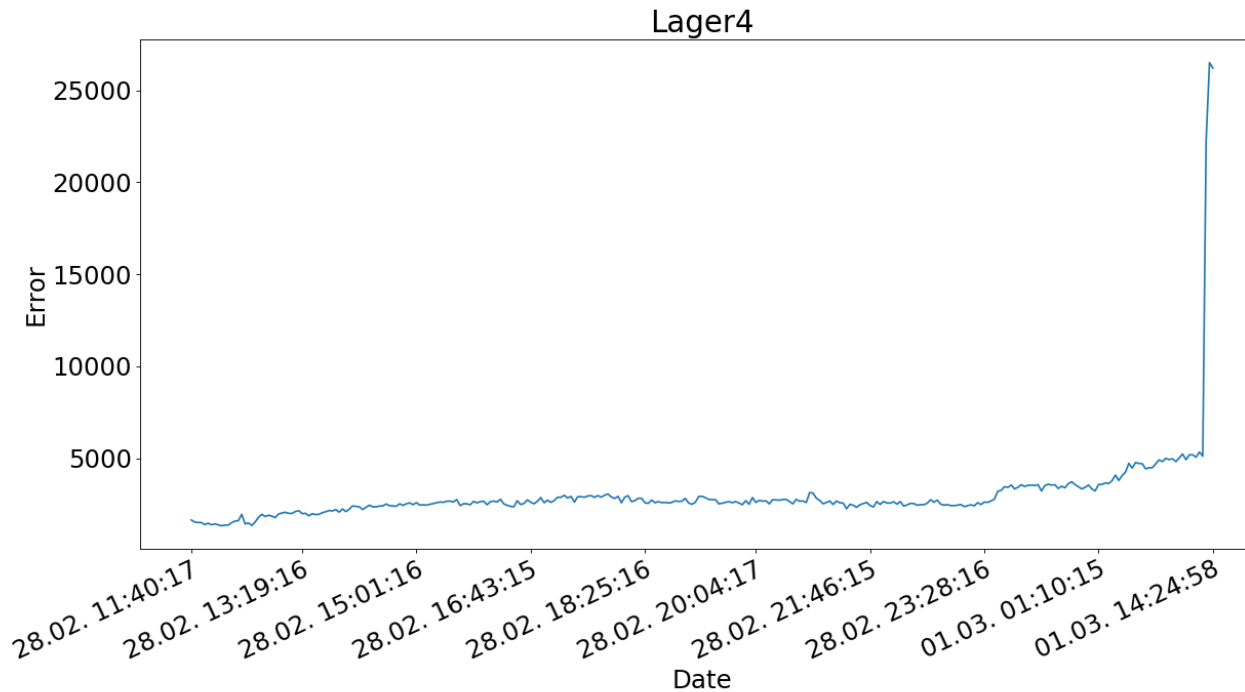
## Bad reconstruction



# Autoencoder Architecture



# Autoencoder Results - Qualitatively



Window size: 128

Latent space size: 50%

- Behaviour of error is as expected: As bearing degrades error goes up
- Error functions has steps showing signs of different error states
- Error curve is also smooth thus making finding a threshold easier
- But: This plot is only for window size = 128. How do other window sizes behave?

# Autoencoder Results - Quantitatively

**Lager 4: AUC depending on window sizes and latent space size**

	128	256	512	1024	2048	4096	6144	8192
50%	0.98	0.83	0.64	0.72	0.75	0.80	0.77	0.79
25%	0.82	0.59	0.47	0.57	0.62	0.75	xxx	0.77
12,5%	0.57	0.51	0.45	0.50	0.03 ?	0.67	xxx	0.69

**Lager 5: AUC depending on window sizes and latent space size**

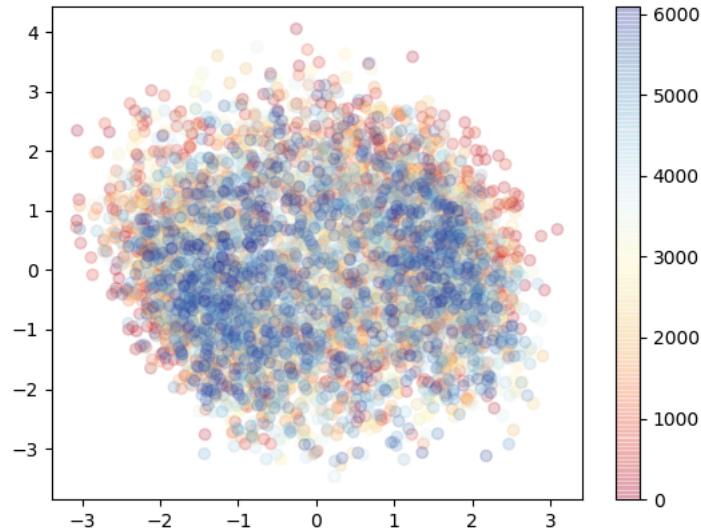
	128	256	512	1024	2048	4096	6144	8192
50%	0.97	0.92	0.91	0.82	0.87	0.70	xxx	0.77
25%	0.94	0.87	0.91	0.93	0.82	0.84	xxx	0.75
12,5%	0.91	0.96	0.72	0.88	0.93	0.84	xxx	0.70

- AUC performance seems to correlate with window size
- Network Lager 4, window size 4096, latent space 12,5%: learned the inverse of the expected behaviour

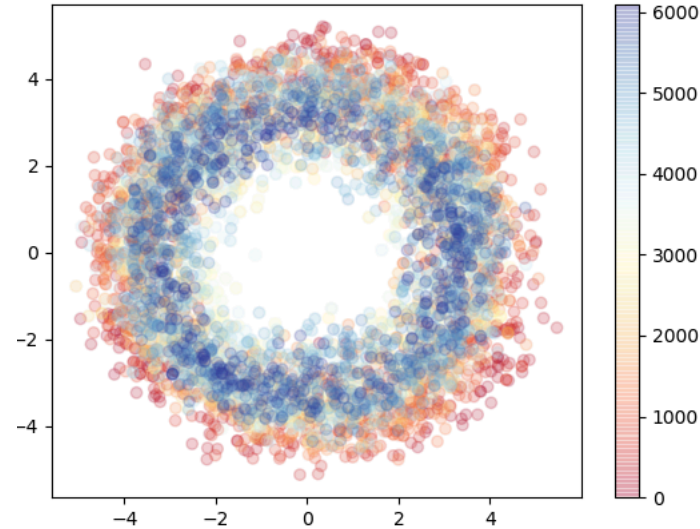
# Autoencoder Results - Latent Spaces

- From previous experiments we expect a clustering of latent spaces when applying PCA or TSNE
- We plot latent spaces transformed by PCA into 2D as the bearing runs (from early = red to late = blue)
- TSNE did not yield any useful clustering results

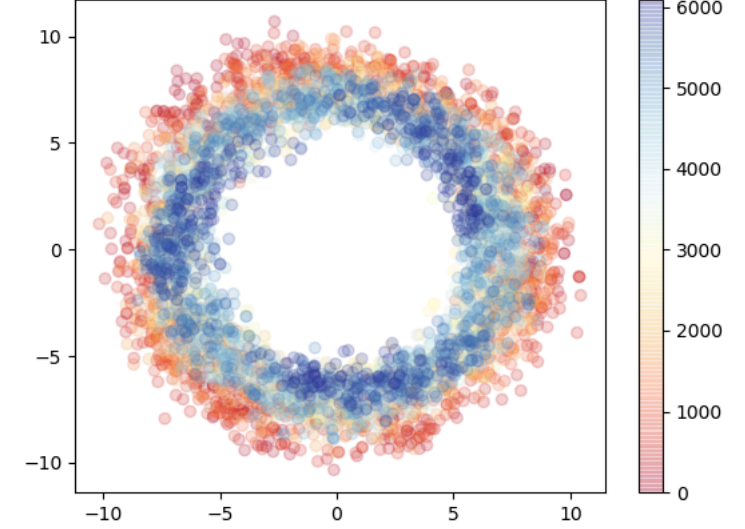
**Lager 4; WS 128; Lat. 50%**



**Lager 4; WS 1024; Lat. 50%**



**Lager 4; WS 4096; Lat. 50%**



- Clustering is better the bigger the window size is → inverse correlation with AUC results

# Autoencoder - Reduced Model

- Window size 128 yielded the best results → reduce the window size further to check if this trend continuous
- Reducing the model to 3 down sampling steps to make window size 8 possible

**Lager 4: AUC depending on window sizes and latent space size**

	8	16	32	64
50%	1.0	0.99	0.98	0.98

- Although the window size is very small the network behaves as expected
- The error noise caused by the degradation introduces high frequency components into the signal
- These components thus also influence very small window sizes

# Conclusion, Outlook and Problems

- Degradation starting point can be determined by an expert exactly knowing which feature to look at
- Finding a classical generally applicable feature is very hard
- The application of an autoencoder machine learning model reduces the feature space to 1 error plot
- Some fine tuning parameters remain e.g. the window size making the model not always universal
- We only evaluate the separability of the resulting error plot
- No instructions yet on how to find a good threshold
- Sometimes the network is better in reconstructing erroneous data (reversed classifier) than "normal" data
- Inverse correlation of latent space clustering and performance