On Model Performance Estimation in Time Series Anomaly Detection

Arn Baudzus
Federal Institute for occupational Safety and Health (BAuA)
Dortmund, north rhein westphalia, Germany
baudzus.arn@baua.bund.de

Bin Li
Department of Computer Science TU Dortmund University
Dortmund, north rhein westphalia, Germany
bin.li@tu-dortmund.de

Adnane Jadid
Federal Institute for occupational Safety and Health (BAuA)
Dortmund, north rhein westphalia, Germany
jadid.adnane@baua.bund.de

Emmanuel Müller
Department of Computer Science TU Dortmund University
Dortmund, north rhein westphalia, Germany
emmanuel.mueller@cs.uni-dortmund.de

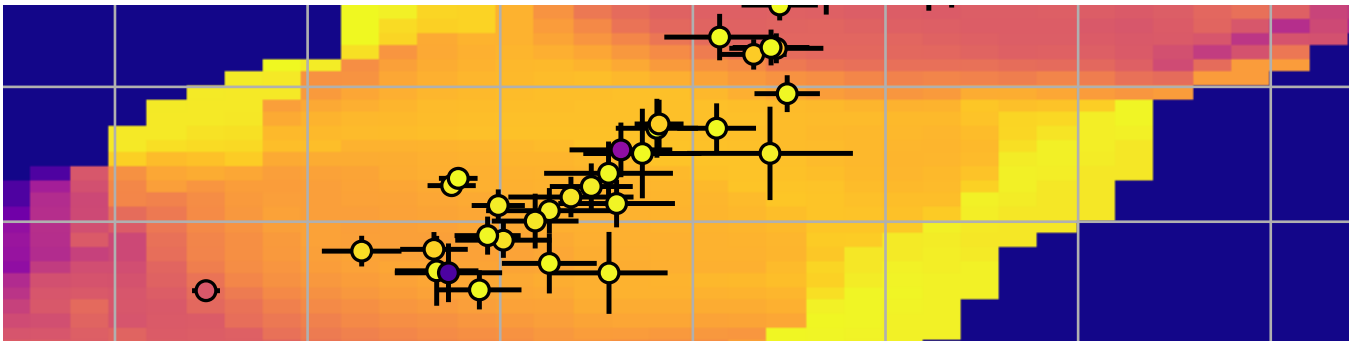


Figure 1: An artistic impression of the generalisation of a NN-classifier that is discussed in this paper.

ABSTRACT

The usual way to quantify the performance of a novel algorithm in the field of classification, especially in time series anomaly detection, is to compare its performance against selected baseline competitors on selected data sets. There is a common sense in the community which data sets and baselines should be considered when evaluating the algorithm's performance. Nevertheless, on which basis data sets and baselines should be selected is frequently discussed.

In this paper, we propose an index for univariate time series data in anomaly detection based on information theory. The index shows an association with the AUC-score of an anomaly detection algorithm that is trained on the data, meaning that, the index can be used as a proxy for the "difficulty" for the classification task, this data set holds.

A workflow to quantify this association using an interpretable classifier that relies on the index and a derived performance baseline is developed. The classifier performs within the margin of error of this performance baseline, meaning that we were not able to clearly mathematically show the association between index and AUC-score.

We believe that our work, which unites mathematical concepts from information theory, physics, and computer science, is innovative and generally points in a promising direction that is worth investigating.

CIIS 2023, November 25–27, 2023, Tokyo, Japan

ACM ISBN 979-8-4007-0906-7/23/11.
https://doi.org/10.1145/3638209.3638226

CCS CONCEPTS

• **Mathematics of computing** → **Information theory**; • **Computing methodologies** → **Machine learning**; • **Applied computing** → *Physical sciences and engineering*.

KEYWORDS

Pre-training performance estimation, performance estimation, machine learning, time series anomaly detection, information theory, entropy, mutual information, Fourier transformation, error propagation

ACM Reference Format:
Arn Baudzus, Bin Li, Adnane Jadid, and Emmanuel Müller. 2023. On Model Performance Estimation in Time Series Anomaly Detection. In *2023 the 6th International Conference on Computational Intelligence and Intelligent Systems (CIIS) (CIIS 2023), November 25–27, 2023, Tokyo, Japan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3638209.3638226

1 INTRODUCTION

Time series anomaly detection is the task of finding points or groups of points in a time series that significantly differ from the distribution of the majority of the points.

Defining this "difference" between the normal and the abnormal state in a strict mathematical sense is, if it is even possible, very difficult. An anomaly could be something like a single value, that is numerically different from the rest of the values in the time series. It could also be something like a break in a reoccuring pattern or a difference in the way two dimension of a time series relate to each other (e.g. a time series containing two signals which are periodic and in phase, that in rare occasion go out of phase).[8]

Since no general definition of an anomaly is known, it is also not feasable to write a classical algorithm for general purpose anomaly detection. Machine learning algorithms however, overcome this requirement of an explicit definition.
In the machine learning context, the information what is normal and what is abnormal is implicitly characterized by the training data. So the data implicitely defines the classification problem at hand.

There are different methods for time series anomaly detection [12]. In this paper, we focus on reconstruction based approaches are reconstruction based approaches. Here, the anomaly detector is based around two core components. One is a machine learning model, the other is a discriminator.

The model is trained to reconstruct **normal** samples from the data. There are diffrent aproaches how this model could look like (Examples included in this work are feed forward networks, recurrend neural networks, convolutional neural networks and attention based network). A common property most reconstructing models share, is an information bottleneck that is realised in the network structure (e. g. an autoencoder structure). This way the model is trained to prioritise information that shall be passed to the bottleneck to construct a faithful recreation of the input. A good quality reconstruction can only be archived when other properties of the data set, that are not passed trough the bottleneck are aggregated in the network structure.
The network "learns" how the normal state behaves.

The heuristic to detect anomalies with this reconstructing model is as follows: The network must "know" the properties of the normal state, to reconstruct it. If an anomaly gets passed trough the network, the reconstruction will not be as good, since the network isn't trained to recreate it.
The second part of the anomaly detector, the discriminator, is an algorithm that processes the error (e.g. L1 or L2 error), the reconstructing model makes. This could either be another machine learning algorithm or a classical algorithm like an adaptive thresholding. This part of the algorithm labels an input, or a certain part of an input abnormal, based on the reconstruction error[1].

Time series anomaly detection is basically a binary classification task (finding out if a value is abnormal or if it is not.). There are also different methods to quantify the quality of a binary classification. In this work, we settled on the AUC-score as performance indicator.

In the field, huge effort is put into the creation of new algorithms for specialized applications or better performance over all.
There are many different types of time series and sometimes its hard to say what makes an algorithm perform well on one data set and badly on another.

There is no real gold standard to evaluate the performance of an algorithm. Usually, the algorithm is compared against others on a variety of data sets. The choice of these data sets (and the classification problems they hold) is up to the authors.
To assert proper testing, the benchmark data sets have to represent the field of the intended use case of algorithm, and the baselines have to be relevant and should represent the state of the art.
Especially for general-purpose algorithms, there is a variety of benchmark data sets the community agrees on, and that are often used ([19] [1] represent research where this workflow is employed.).

To formalize this methodology, some authors, e. g. Dau et al. [4] collected a variety of data sets with the aim to cover most of the real-world time series that might be faced in the industry.
Other authors like Lai et al. [8] explored the possibility to formally define mathematical criteria that describe a data set. They developed a taxonomy for different types of anomalies.

Our work follows this approach as well. The primary issue with the taxonomy Lai et al. created is that, while it is well suited as a foundation to develop data generators, it is hard to apply to existing data [2]. In this work, this issue is addressed by using only formalisms that require very few assumptions on the time series data to cultivate a computational measure that is generally applicable. The second requirement for this measure is that it should be straightforward to calculate.

To fulfill these criteria, concepts from information theory are used. Entropy and mutual information are concepts that are generally applicable since they require very few assumptions about the system in question and are straight forward[3] to compute.

This paper is structured in two parts.

First, a way to assign an index to a time series data set is proposed. The computation is motivated by information theoretical considerations.
Furthermore, this index is calculated for a variety of different data sets. Following this step, several reconstruction-based anomaly detection algorithms are trained on the sets. The relation of the index to classifier performance is evaluated graphically.
For the classifiers at hand, it is visually notable that the index is "weakly associated" with the classifier performance.

Second, to further investigate and quantify this "weak association", a nearest neighbor-based classifier that outputs an estimated performance of the trained algorithm is developed.
The idea here is that the nearest neighbor classifier relies on the same information visible when looking at the plots discussed earlier but offers a mathematical way to discuss and further evaluate this information. Since the classification and the uncertainty of the classifier are only dependent on the index and the anomaly detection algorithms results, they can be used as a proxy to quantify the

[1] An exception to this are some attention based approaches, that are still trained to reconstruct the input, but to classify if the input is abnormal or not, not only the reconstruction, but also the behaviour of the attention mechanism is evaluated.

[2] The definition of collective outliers e. g., relies on shapelet functions. If these functions are known, like they are in a generative context, the criteria at hand are straightforward to verify. Finding these functions for an existing time series might prove difficult.

[3] There are some caveats that will be addressed in section 2.

quality of the index[4]. We introduce two baselines to compare the algorithm against. One is a classifier that is just randomly guessing, the other one is a classifier that always outputs the average of the measured AUC-scores. The latter one is the important baseline to surpass. Surpassing this baseline means that further information that are not present in the average of the AUC-scores had to be used, which are held in the index.
While the classifier outperforms the baseline of a classifier that is just randomly guessing, it performs within the margin of error of a classifier that always returns the average of all recorded performance results of the algorithm.
It is further investigated how the nearest neighbor algorithm generalizes and how the uncertainty of the algorithm is distributed in the parameter space.
Additionally, it is discussed how the distributions of the predictions of the nearest neighbor classifier compare to the distribution measured values.
Finally, all points and variables that could be alternated to further investigate are listed.

The main contributions and novelties of this work are:

- The concept of the FEMI-index is introduced. The FEMI-index represents a novelty. To our knowledge, although information theoretical influences are present in the field of time series anomaly detection, there is no work using similar concepts to describe the properties of data sets.
- The workflow described in this paper shows how the "high level" concept of the index containing information on the difficulty of the anomaly detection task that comes with a data set could be quantified: In this paper, an interpretable classifier with uncertainty estimation is developed to serve as a tool to quantify the influences of the index at hand. Additionally, two essential baselines are motivated. Surpassing both baselines is strong evidence that the index under test actually contains information about the difficulty of the anomaly detection task for a given algorithm.
- The uncertainty quantification in this work applies Gaussian error propagation[5] (introduced in section 4.1) to topics in time series anomaly detection.

2 FOURIER ENTROPY MUTUAL INFORMATION (FEMI)-INDEX

First, the concepts of entropy and mutual information are briefly presented. In addition, an introduction to the intuition behind the computation of the Fourier entropy mutual information (FEMI)-index is given. A mathematical formulation is stated at the end of this chapter.

[4]This is kind of the same as the relation between correlation and regression. A linear regression algorithm takes the data in and delivers a linear function describing the data. Although when doing correlation analysis, one is only interested in the goodness of the fit as a proxy to investigate how well the data fits a linear function.
[5]Gaussian error propagation is a standard for quantifying uncertainty from experimental physics.

2.1 Entropy and Mutual Information

The entropy[6] is a measure for the expected statistical information of a continuous distribution of values. Let $x \in X$ be a random variable with values in X. Let $p(x) : X \mapsto [0, 1]$ be the probability density function of x. The (differential) entropy of x is defined as:

$$h_x = -\int_X p(x) ln(p(x)) \mathrm{d}x \tag{1}$$

Mutual Information is a measure for the difference in statistical information in one distribution compared to another. For two random variables x and y, the mutual information is defined as:

$$I_{x,y} = h_x + h_y - h_{x,y} \tag{2}$$

It vanishes exactly when both distributions are independent.

In real-world scenarios, it is often not possible to obtain $p(x)$, instead, one is often left with samples
$S := \{x_i \in X, i \in \{1, \cdots, n\} \subset \mathbb{N}\}$ that are assumed to be drawn from the distribution of x. For this case, there are methodologies to estimate the entropy and mutual information of x from the samples x_i. The straight forward approach for estimating the entropy from a set of samples is to bin the samples. The ratio of the number of samples in one bin and the bin size is an approximation to the underlying distribution. The quality of these estimates varies with the applied binning strategy. Best values can be expected with an adaptive binning strategy that is based on the samples.
For one dimensional data, it is also possible to sort the samples. The distance between one sample and it's neighbour is inverse proportional to the distribution function. With this intuition, an estimate for the entropy can be derived. There are also generalisations of this method relying on ranking the neighbouring points based on some distance function.
Once the entropy is known, the mutual information can be calculated using equation 2. We refer to Kraskov et al. [7] for a deeper insight into the calculation process.

In this work, an entropy sampler developed by Marín-Franch et al. [10] is used. The entropy and mutual information that are estimated from sets of samples S_1 and S_2 is denoted as $h(S_1)$,$h(S_2)$ and $I(S_1, S_2)$.

2.2 The Intuition behind FEMI-Index

To compute the index, a data set is needed that is split into two subsets. One subset contains only data that is considered normal. One contains normal data alongside abnormal data.
Note that the data itself doesn't have to be labeled.
The goal here with the index was to find a small number of mathematical properties that describe the data set. The following bullet points introduce the parameters used and the intuition behind the inclusion of these:

- By calculating the entropy of the subset containing only the normal data, it is asked: "How much information is there to be learned to quantify the normal state of the system that is described by the data?".

[6]In our work we use the differential entropy. The differential entropy formally looks like a generalized form of the Shannon entropy for discrete distribution, which it isn't. It still is a meaningful statistical property of a distribution.

- By calculating the mutual information between the subset containing normal data and the subset that contains abnormal data, it is asked: "How much information is there separating the abnormal case from the normal case?"

However, computing these measures just straight up on the data would implicitly ignore the time series nature of the data since the order in which values are recorded would be ignored.
To tackle this issue, the Fourier transform is taken, and the triplets of radius and angle of the complex amplitude and the corresponding frequencies from the Fourier transformation are aggregated. The information theoretical measures for the distribution of those triplets is calculated.
The whole process of calculation is visualized in a flow chart, seen in figure 2.

2.3 Formal Description

To formally describe the calculation of the FEMI-index, a definition of time series data is introduced:
For each data set, let $I = \{1, \cdots, n\} \subset \mathbb{N}$ be a set of indices. For every $i \in I$ there is $t_i \in \mathbb{R}$ and $x_i \in \mathbb{R}$. t_i are called timestamps and x_i are called values. In all of the data sets used in this work, the values were recorded at equidistant points in time. Thus the timestamps t_i are ignored in the further description of the formalism (They had to be included otherwise.).

For a constant $m \in \mathbb{N}$ (the window size) and some $j \in (1, \cdots, n-m)$ let $(x_j, x_{j+1}, \cdots, x_{j+m}) := s_j \in \mathbb{R}^{m+1}$ be a data set entry.

In the following paragraphs, the computation of the FEMI-Index is introduced step by step:
To calculate the FEMI-Index, the entries are first multiplied by the Hanning window function. This way, artifacts that would emerge from the non periodicity of the values of time series represented by s are prohibited. The Hanning window for an entry with n values is given by:

$$w(j) = \frac{1}{2}(1 - cos(2\pi\frac{j}{n})) \quad 0 \leq j \leq n \tag{3}$$

After windowing, the Fourier transform of the entry s is calculated. In this work, the following version of the discrete Fourier transform is used. The Fourier coefficient for the frequency k is given as:

$$F(s)_k = \sum_{l=0}^{m-1} e^{-2\pi i \frac{kl}{m}} x_l \tag{4}$$

In this formula, the indices for s_j are shifted from $j, \cdots, j+m$ to $0, \cdots, m$.

The complex output of the Fourier transform then has to be transformed to the $\mathbb{R}^2$. This is done by either taking the radius and angle of the complex amplitude or by taking its real and imaginary part. Each variant leads to a different version of the FEMI-Index.
This way, each set s_j is translated to a set f_j containing entries $f' \in \mathbb{R}^3$. f_j is a triplet, the first two values represent the complex amplitude from the Fourier transform the third element is the corresponding frequency.
For each data set, two sets of entries are needed. One where the entries contain anomalies (in a rate that is expected for the data source that underlies the set) and one without anomalies. Formally written: For a set size $p \in \mathbb{N}$, Two tuples[7] $i_n, i_a \in (1, \cdots, n-m)^p$ are needed such that two tuples of entries can be defined:

$$D_{\text{normal}} = (s_j : j \in i_n) \quad D_{\text{abnormal}} = (s_j : j \in i_a) \tag{5}$$

D_{normal} contains the normal enties, D_{abnormal} can contain abnormal entries.
By applying the above-mentioned procedure of multiplying the Hanning window, Fourier transforming and converting the entries in each tuple, new tuples are obtained:

$$D'_{\text{normal}} = (f_j : j \in i_n) \quad D'_{\text{abnormal}} = (f_j : j \in i_a) \tag{6}$$

By concatenation:

$$S_{\text{normal}} = \bigcup_{f \in D'_{\text{normal}}} \bigcup_{f' \in f_j} f' \quad S_{\text{abnormal}} = \bigcup_{f \in D'_{\text{abnormal}}} \bigcup_{f' \in f_j} f' \tag{7}$$

Two sets are obtained, each containing three-dimensional entries. The FEMI-index is then obtained by computing:

$$(E, MI) = (h(S_{\text{normal}}), I(S_{\text{normal}}, S_{\text{abnormal}},)) \tag{8}$$

3 FEMI-INDEX CALCULATION

In this chapter the conduction of the experiments is reported. 15 models where tested on 128 datasets. As a measure of anomaly detection performance, the AUC-score was computed. For each dataset, the FEMI-index was computed as well, resulting in the data points of FEMI-index and AUC-score per data set, that are used trough out the rest of the paper. The error estimation is described as well.

The FEMI-index was evaluated on several benchmark data sets. The benchmark data consists of the following sets:

- synthetically generated sine-waves with anomalies, where the anomalies manifest either in a deviation from the normal periodicity of the sine or its Amplitude (4 data sets).
- The datasets from the UCR-archive [4] are used. Since the UCR-Archive is a time series classification data set[8], we labeled one class as an anomaly and the others as normal (121 data sets).
- Data from machine 1-1 from the SMD-Dataset [13] is used. The data is converted into univariate data by looking only at the dimensions where the anomaly manifests itself (2 data sets[9]).
- Data from the ECG data set [5] is used (1 data set).

We randomly sample Examples from these data sources and group them to obtain the sets needed for the FEMI-index calculation. This stochastic nature of the data sources also affects the computed FEMI-index. The indices reported are the mean of 10 calculations

[7] We chose tuples instead of sets to give the possibility to include indices multiple times. We still use set notation, though.
[8] There is also a UCR archive for time series anomaly detection, which we did not use in this work.
[9] We originally planned to use more examples from SMD but failed to do so due to an error that emerged in the experiment code.

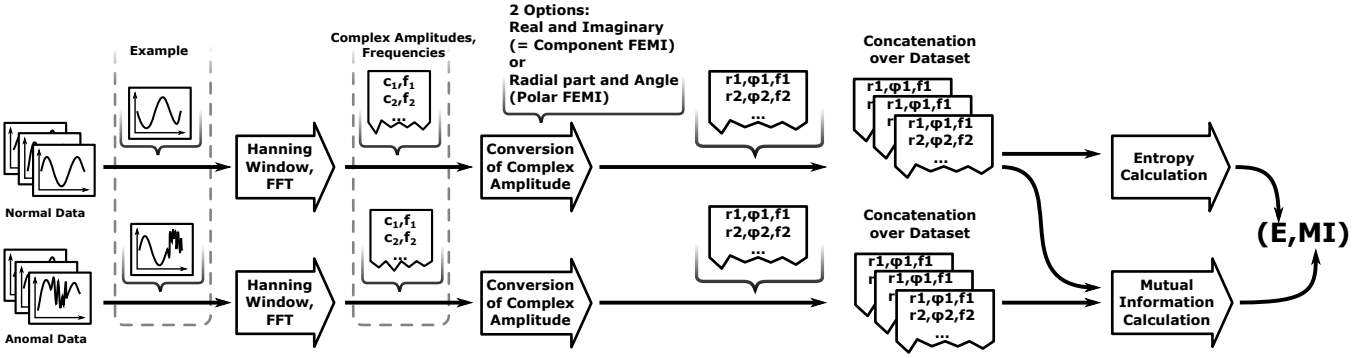


Figure 2: This flowchart visualizes the process of computing the FEMI-index. Note that there are two variants of the computation, depending on whether one chooses the component or the polar transformation of the complex amplitudes into the $\mathbb{R}^2$. The variables c_1 and c_2 represent the complex amplitudes that correspond to the frequencies f_1 and f_2. The variables r_1,r_2,ϕ_1 and ϕ_2 are the radius and angle of c_1 and c_2.

for each data source. The errors seen in the plot are the standard deviation for these calculations.
Taking this error into account makes the evaluation of the index more "realistic" since sampling data from a system of interest in multiple instances would cause a similar variation of the index.

To further evaluate the association between the index and the anomaly detection algorithm's performance, models are trained on each of the data sources. As of now, we limit ourselves on reconstruction based anomaly detection methods. Simple "toy model like" implementations are applied in this work instead of state of the art algorithms, since most state of the art algorithms consist of very sophisticated data augmentation strategies and loss functions working in conjunction. By taking a simpler approach, we hoped to find behavior that can be associated with the model structure, which would be harder to find when comparing more complex models.
All systems have in common that they are trained to reproduce the input data. The reconstruction error of the input is used as a proxy for the anomaly of the data point. For all models, the AUC-Score on the data sets was recorded. Different models were tested and are described in the following section. Additionally, the models referenced in table 1 are introduced. All models are implemented in `pytorch` [11].

Feed Forward Autoencoder (0-3). These models are feed-forward autoencoders with ReLU activation function. The latent space is L times smaller than the input and is always on the middle layer (4) in this case. On the other layers, the number of neurons per layer is linearly decreasing/increasing to map the input to the latent space and the latent space to the output.

CNN Autoencoder (9-12). An autoencoder that consists of convolutional layers which are connected by linear layers which mediate the change in dimension of the information that is passed form layer to layer. In addition to the "normal" encoder, which operates on the time domain and processes the time series, a second encoder can be added, either alongside the first one or instead of the first one, that processes the Fourier transformed version of the time series (Following an idea shown in a blog post by Ilia Zaitsev [18]).

Table 1: This table lists the different models that were tested and their corresponding IDs. FF-AE stands for feed forward autoencoder, CNN-AE for convolutional neural network autoencoder. A more in-depth description of the parameters and a link to the implementation that was used can be found in the test.

Model ID	Model Type	Comment
0	FF-AE	7 Layers, L = 0.1
1	FF-AE	7 Layers, L = 0.25
2	FF-AE	7 Layers, L = 0.5
3	FF-AE	7 Layers, L = 1
8	CNN-AE	-
9	CNN-AE	w. FFT
10	CNN-AE	FFT only
11	CNN-AE	w. FFT, no Han.
12	CNN-AE	only FFT , no Han.

Before the transformation, the time series could either be windowed with a Hanning window or transformed directly.

In addition to these models, we originally also included LSTM / GRU based reconstruction based models (IDs 4-8) and transformer-based approaches (IDs 13 and 14) in the testing. Unfortunately, there were errors in the code that were discovered after the benchmarking, rendering the benchmark results unusable. In the transformer code, it is just simply due to a bug[10]. The recurrent neural networks need an optimizer that is more specifically tailored to the task. The loss during the training epochs indicate that there are stability issues. We thus decided to exclude these models from the discussion. For the curious reader, we included additional information, example plots, and results from these models in the appendix (section A.2).

All of the models where trained for 40 epochs using the Adam-optimizer [6] with a learning rate of 0.001. Convergence was confirmed by evaluating the loss functions for a portion of the models.

[10]A final layer for the scaling of the output is missing. Hence the output can only reach a range between -1 and 1, which didn't show during the testing, since testing was done on normalized data.

The specific implementations can be seen in our git-hub repositories[11] The AUC-score has an uncertainty that covers the errors made in the numerical integration to obtain the AUC-score. In our computation, the ROC-curve of the anomaly detection algorithm is sampled by equally spaced thresholds from 0 to the maximum of the anomaly score. To obtain the AUC, a trapezoid quadrature is used. The error for the quadrature is the difference between the AUC estimated by an upper and lower right-hand rule. Geometrically, this exactly covers the uncertainty that is introduced through the sampling process, but it doesn't include errors that might be present in the values for the ROC-curve, which we assume to be smaller than the error that occurs from sampling. The errors are usually below 0.05 but especially the SMD set shows AUC-scores around 0.5 and errors around 0.4, which points towards a sampling problem. This can be assessed by using an adaptive quadrature.

For simplicity, model 3 and model 10 are chosen for the discussion of the FEMI-index. Model 3 is an instance of the feed-forward type networks, and model 10 is an instance of the convolutional networks. The other networks of these types performed similarly.

Figure 3 shows the component and the polar FEMI-Index for model 3 and model 10.

The entropy and mutual information reported here are negative. This should not be possible. We suspect, that this is due to an error in the experiment code. We still report all findings as is, since the results are reproducible[12] and the empirical relations we discuss here still hold.

Comparing the top two images (3a, 3b) to the bottom two images (3c , 3d) it is noticeable, that the component FEMI-index is much wider spread in the parameter plane, whereas the polar indices are mainly gathered around a "line shaped" region in the center.

The figures show a connection between the FEMI-index of the data set and the AUC-Score of the algorithm in some regions of the plots. In the top right region of the plots, no clear trend in the archived AUC-Score is visible on those sets. AUC-Scores that range from near 0 to near 1 are all present in that region. In contrast, on data sets that are indexed in the bottom left of the plot, the algorithm generally archives higher, more consistent AUC-Scores. This holds, except for a few deviating data set in that region. This trend is best grasped in the plot of the polar FEMI-index (figures 3c and 3d).

For the rest of the paper, visualizations of the polar FEMI-index are used since the phenomena that are discussed are easier to grasp visually in those plots.

4 NEAREST NEIGHBOUR (NN)-CLASSIFIER

In this section, a nearest neighbor algorithm that gives an estimation for the performance of the classifier based on the FEMI-Index and AUC-Scores discussed in the last section is applied. This classifier fulfills two purposes:

Firstly, this classifier is a way to more precisely investigate the phenomenon that the AUC-Score, for some indices, is associated with the FEMI-Index of the data set.

[11]The Experiments can be found here: https://github.com/Arn-BAuA/FEMIIndex The Model implementations and a indepth documentation of all models can be found here: https://github.com/Arn-BAuA/TimeSeriesAEBenchmarkSuite.

[12]Meaning, that the numbers we are computing also still function as a number to identify the data.

Second, finding a good performing classifier for one algorithm means that it would be possible to get an estimate for its performance. This estimate for a novel data source is based on the performance measured when benchmarking the algorithm on known data sources by calculating the FEMI-Index of the data source and query the classifier. This pre-training estimate saves computation time and improves results when combined with e. g. an ensemble method.

Nearest neighbor algorithms are a standard tool in machine learning [15]. In this case, a nearest neighbor classifier was chosen for its simplicity and interpret-ability since the performance of the classifier has to function as a proxy to further measure the quality of the FEMI-index.

In this section, the steps of the algorithm are presented. Due to the importance for the next steps, a small introduction to Gaussian error propagation is given. This chapter is concluded by a formal description of the nearest neighbor algorithm.

The nearest neighbor algorithm presented here works as follows:

- The user inputs a point (E, MI) in the FEMI-Plane.
- The user specifies a radius (it is a hyperparameter). A radius was chosen instead of a number of nearest neighbors since the interpretability in less populated points of the domain seems to be easier.
- The algorithm outputs the average of the AUC-Scores of known data points that fall into a circle of the user-specified radius with the user-specified point at its center.
- This average is weighted with a function that accounts for the distance of the points (points that are closer to the center of the circle should have more influence on the output.)
- Additionally, a measure for the uncertainty is computed by calculating the propagation of uncertainty (Gaussian error propagation) for the AUC-Score, the entropy, and the mutual information.
- The error obtained by propagating the uncertainty is compared to the standard deviation of the AUC-scores in the circle. The maximum of both is chosen.
- If no point is in the circle, an AUC Score of 0 ± 0.5 is returned.

Since Gaussian error propagation is an essential part of the uncertainty quantification of the classifier, the next section provides a short introduction.

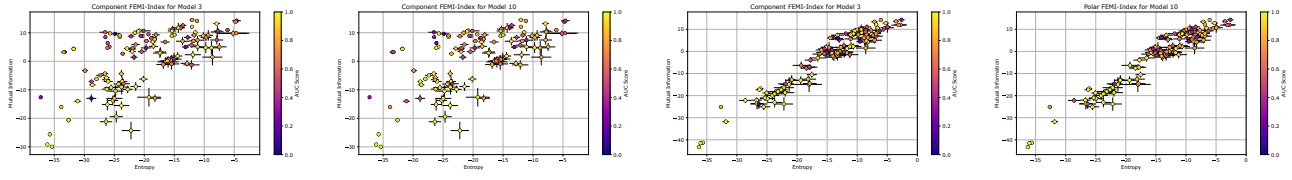
4.1 Gaussian Error Propagation

While Gaussian error propagation is a standard in propagating errors in calculations done in experimental physics, it is relatively uncommon in computer science. Hence, this section provides a short introduction since Gaussian error propagation plays a key role in the classification of uncertainty that is done in the next sections.

There are many ways to derive Gaussian error propagation (e. g. from the Taylor series). In this introduction, the formula of the error propagation is graphically motivated.

The task of error propagation is as follows:

Given a function $f : X \mapsto Y$ and a value $x \in X$ with an associated uncertainty $\Delta x \in X$. For a given value $y = f(x)$, the task of error propagation is to estimate $\Delta y \in Y$, the error of the calculated value y.



(a) Component FEMI Index for Model 3 (b) Component FEMI Index for Model 10 (c) Polar FEMI Index for Model 3 (d) Polar FEMI Index for Model 10

Figure 3: This figure shows the two components of the FEMI-Index of multiple datasets. In addition, the AUC-score that was reached by an anomaly detection algorithm that was trained on the data is shown in the color index. The left two images (a and b) show the component FEMI-index, and the right two images (c and d)show the polar FEMI-index. The images a and c show the results for model 3, a feed-forward network-based model. Images b and d show the FEMI-Index for a convolutional network-based model (model 10). An association between the position of the FEMI-index and the AUC-score can be seen, especially in picture c. The entropy and mutual information in the plots are negative. This is due to an error in the experiment code. The numbers reported here still function as index to identify the data sets. As the results discussed here are of empirical nature, we continue the discussion with these values.

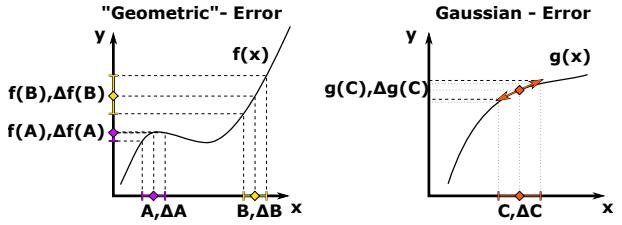


Figure 4: This figure graphically shows the task of error propagation. The errors of $f(A)$ and $f(B)$ are derived graphically on the left-hand side. The right-hand side plot shows a graphically conducted Gaussian error propagation compared to the geometric error (dotted line).

For a one-dimensional function f the errors can be found graphically as shown in figure 4.

The left hand side of figure 4 shows geometrically how the error of the values A and B effects the uncertainty of the outcomes of f. For a one-dimensional function, this geometric error propagation is easy to draw, but it is not straight-forward to do analytically. Notice how even for this rather simple function f presented in the figure 4, the maximum value of the interval around $f(A)$ is not explicitly known (meaning that it is some unknown value for an x' between A and $A + \Delta A$.).

Gaussian error propagation can be seen as an approximation to this geometric error propagation. Instead of searching for the values that characterize the intervals of the geometric error distribution, one linearly approximates the function at the point where it is evaluated, and then calculates the interval bounds of the geometric error propagation for this linear approximation, which can be done analytically. This is depicted on the right-hand side in figure 4.

Formally this writes out as follows: The deviation in the output space from the value $f(x)$ can be (in the one-dimensional case) written as:

$$d = \underbrace{\frac{\partial f}{\partial x}(x)}_{\text{The linear Approximation.}} \cdot \underbrace{\Delta x}_{\text{The strength of displacement.}} \tag{9}$$

The estimated error then is the Euclidean norm of this deviation.

$$\Delta y = \sqrt{(\frac{\partial f}{\partial x}(x) \cdot \Delta x)^2} \tag{10}$$

In a multidimensional way, where the (scalar) output y depends on multiple input variables x_i this Gaussian error propagations is the canonical multidimensional equivalent of this geometric intuition. Meaning, the error x and Δx are multidimensional vectors. Instead of a simple derivative, the gradient of y is used, and instead of the norm of the deviation, the following term is computed:

$$\Delta y = \sqrt{\sum_i (\frac{\partial f}{\partial x_i}(x) \cdot \Delta x_i)^2} \tag{11}$$

Geometrically interpreted, this is the Euclidean length of the gradient of y at the point x that was dimension-wise scaled by the entries in Δx.

4.2 Formal Description

The section is devoted to present the nearest neighbor classifier. Let $D_j = (D_{j\,\text{normal}}, D_{j\,\text{abnormal}})$ for $j \in (1, \cdots, N) := J$ be a number of data sets. For every one of these sets, a FEMI-index (E_j, I_j) has been calculated.

To define the output of the classifier, some functions have to be defined:

On the plane of FEMI-index values, a distance measure $d((E_1, MI_1), (E_2, MI_2)) : \mathbb{R}^2 \mapsto \mathbb{R}^+$ is used. In this work, we chose this distance to be the Euclidic distance.

$$d((E_1, MI_1), (E_2, MI_2)) = \sqrt{(E_1 - E_2)^2 + (MI_1 - MI_2)^2} \tag{12}$$

For a constant radius $R \in \mathbb{R}^+$ the set $B_R(E, MI)$:

$$B_R(E, MI) := x = (x_1, x_2) \in \mathbb{R}^2 \text{ such that } d((x_1, x_2), (E, MI)) \leq R \tag{13}$$

is defined. Its the set of points in $\mathbb{R}^2$ that have a distance d that is smaller or equal than R from (E, MI).

With these functions, for a given classifier, an AUC-score AUC_j for set D_j was calculated as well. The classifiers estimate for the AUC-score at a given point (E, MI) is defined as:

$$AUC_C(E, MI) = \sum_{\substack{j \in J \text{ such that:} \\ (E_j, MI_j) \in B_R(E, MI)}} AUC_j \frac{w_R(d((E, MI), (E_j, MI_j)))}{W_R(E, MI)} \tag{14}$$

In this equation w_R is a weight function and W_R is its normalization. The weight functions are meant to give the possibility to weight points in the FEMI-plane that are near the center of the circle stronger than others. Formally we chose:

$$w_R(d((E_1, MI_1), (E_2, MI_2))) = \frac{\sqrt{R^2 - d((E_1, MI_1), (E_2, MI_2))}}{R} \tag{15}$$

The normalization can be calculated as:

$$W_R(E, MI) = \sum_{\substack{j \in J \text{ such that:} \\ (E_j, MI_j) \in B_R(E, MI)}} w_R(d((E, MI), (E_j, MI_j))) \tag{16}$$

To obtain a measure of uncertainty for the classification, Gaussian error propagation is used, assuming that E_j,MI_j, and AUC_j have an error. As discussed in section 3 there are areas in the FEMI-parameter space where no clear association between FEMI-index and AUC-score can be seen. To account for these areas in the uncertainty quantification, the maximum of the error computed following the error propagation and the standard deviation of the AUC-score values in $B_R(E, MI)$ is taken as final uncertainty for the prediction.

5 EVALUATION OF THE NN-CLASSIFIER

In this section, the performance of the nearest neighbor algorithm is evaluated. The section is divided into two subsections. In the first section 5.1, the classifier is compared to the two baselines. This serves two purposes. For one, surpassing these baselines is a statement of the quality of the classifier. Second, and more important: The only way the classifier can surpass the second baseline, which just returns the average of all benchmarked AUC-scores, is to utilize the information that is not present in the average. Since this information is drawn from the FEMI-index, it follows that the index indeed carries information on the difficulty of the anomaly detection task for that given anomaly detection algorithm.

In the second subsection, the distribution of the classifier outputs is compared to the measured distribution (section 5.2). It is also discussed how the classifier generalizes and estimates uncertainty (section 5.3). Those sections are more focused on the properties of the classifier than the FEMI-index.

Evaluation of the nearest neighbor algorithm is done by removing one point from the distribution of points and then predicting it's associated AUC-Score from its FEMI-index using the remaining distribution.

This is done for every point in the distribution. The mean squared error (MSE) of the AUC-Score values is used to assess the performance of the classifier.

$$MSE = \sum_{j \in J} (AUC_C(E_j, MI_j) - AUC_j)^2 \tag{17}$$

Table 2: This table shows the MSE of our two baseline classifiers and the nearest neighbor classifier for different values of R for the polar (P:) and component (C:) FEMI-index for models 3 and 10. All uncertainties are obtained by error propagation.

Error/Model ID	3		10	
Random C. MSE	0.208 ± 0.018		0.212 ± 0.011	
Avg. C. MSE	**0.055 ± 0.020**		**0.048 ± 0.026**	
NN-C. MSE	P:	C:	P:	C:
R = 1	**0.091 ± 0.288**	**0.100 ± 0.307**	**0.088 ± 0.283**	**0.106 ± 0.316**
R = 1.5	**0.087 ± 0.273**	**0.092 ± 0.283**	**0.071 ± 0.248**	**0.083 ± 0.265**
R = 2	**0.063 ± 0.220**	**0.077 ± 0.240**	**0.052 ± 0.195**	**0.067 ± 0.225**
R = 3	**0.055 ± 0.179**	**0.070 ± 0.200**	**0.043 ± 0.153**	**0.050 ± 0.182**
R = 5	**0.052 ± 0.122**	**0.058 ± 0.150**	**0.042 ± 0.116**	**0.045 ± 0.121**
R = 10	**0.048 ± 0.104**	**0.050 ± 0.111**	**0.041 ± 0.095**	**0.042 ± 0.099**
R = 100	**0.055 ± 0.119**	**0.055 ± 0.120**	**0.048 ± 0.109**	**0.048 ± 0.109**

5.1 Comparison to performance baselines

To further get a sense of the performance of the classifier, the classifier's MSE are compared to the MSE of two baseline competitors. The first one is a classifier that just outputs uniform distributed random values between 0 and 1. Surpassing it means essentially that the classifier at hand is better than random guessing. It follows that some information on the performance of the algorithm in anomaly detection is obtainable from the FEMI-index and the AUC-data.
Its MSE can be calculated using the expected values for the contribution to the MSE of the individual data sets, which can be calculated as:

$$\langle MSE_{j\ \text{random}} \rangle = \int_{\text{minimal AUC}}^{\text{maximal AUC}} p(AUC_{j\ \text{random}} = x)(AUC_j - x)^2 \mathrm{d}x \tag{18}$$

Since we assume uniform random distribution of predictions x, $p(AUC_{j\ \text{random}} = x)$ evaluates to $1/(\text{maximal AUC} - \text{minimal AUC})$. The minimal and maximal values of the AUC are 0 and 1, so the integral simplifies to:

$$\langle MSE_{j\ \text{random}} \rangle = \int_0^1 (AUC_j - x)^2 \mathrm{d}x = AUC_j^2 - AUC_j + \frac{1}{3} \tag{19}$$

and with that, the expected value for the MSE of the random classifier is:

$$\langle MSE_{\text{random}} \rangle = \sum_{j \in J} \langle MSE_{j\ \text{random}} \rangle \tag{20}$$

The other is a classifier that just outputs the mean of all measured AUC-Score values. This classifier is an important baseline since, to outperform it, an association between the AUC-Score and the data based on the FEMI-index has to be utilized. Surpassing the performance of this baseline is a strong indicator that the index contains some relation between the difficulty of the anomaly detection task and the data.
Its MSE can be calculated as:

$$MSE_{\text{average}} = \frac{1}{|J|} \sum_{j \in J} (AUC_{\text{average}} - AUC_j)^2 \tag{21}$$

The results of the comparison are listed in table 3. Results for all trained classifiers are listed in the appendix A.1. In all cases, the MSE of the random classifier is surpassed by the nearest neighbor classifier, regardless of the radius.

However, the classifier fails for all radii to surpass the baseline of the average classifier. In some cases, the nearest neighbor classifier

archives a smaller MSE than the average classifier, but only by 0.001, which is a very small difference compared to the uncertainty. For all that we can tell, both classifiers perform equally.

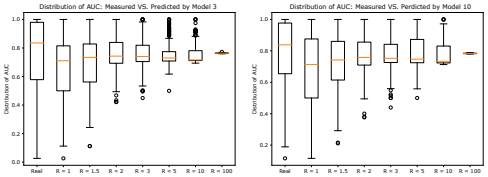
At huge radii, the nearest neighbor algorithm gives the same output as the average classifier, which is to be expected, since the average classifier is "the asymptotic limit" ($R \to \infty$) of nearest neighbor algorithm.

The absence of a measurable lower error compared to the baseline shows that, at least in the investigated example, the classifier could not benefit from the information that is provided by the FEMI-index[13].

Partially, the results of the classifier had lower MSE than the average classifier baseline. However, these numerical values cannot be interpreted since the improvement is small compared to the errors of the values.

5.2 Comparison of AUC-distributions.

To compare the distribution of the predicted AUC-scores and the measured AUC-scores, the box plots of the distributions are used. They are depicted in figure 5 for model 3 (figure 5a) and model 10 (figure 5b).



(a) Boxplot for model 3 on polar FEMI-Index **(b)** Boxplot for model 10 on polar FEMI-Index

Figure 5: This figure shows box plots of the distribution of the AUC-score values that were measured during the benchmark and that were predicted by the classifier using different radii.

For both boxplots, it can be seen that the bulk of the values predicted by the classifier (median and percentiles) is at a lower AUC-score than the original distribution of the values. That is due to the large section depicted in the upper right of the FEMI-index plots (figure 3), where an association between FEMI-index and AUC-Score is not noticeable. In this region, the average calculated by the classifier tends to be around 0.5.

The best values, MSE-wise, are archived by choosing a radius of 5 or 10. With this radius, the classifier holds the ideal balance between taking into account the local features of the index and having enough points for the average calculation to compensate values that deviate from the local trend.
In both plots, a narrowing of the distributions due to the calculation of the average is visible, resulting in the final distribution for a radius of 100, where every prediction is basically the average of the distribution.

[13]Which does not mean that the index does not hold that information.

5.3 Generalisation

To see how the algorithm generalizes, it is evaluated across the whole region, that can be seen in the plots shown in section (3). The prediction, as well as the uncertainty for that prediction, are mapped in the 2D plane of the FEMI-index. These plots can be seen for model 3 and 10 for the polar FEMI-index and a radius of 10 in figure 8.

As can be seen, comparing the generalized predictions for model 3 and model 10 (figure 6a and figure 6b) the generalization of the classifier shows the local association of the AUC-Score and the FEMI-Index.

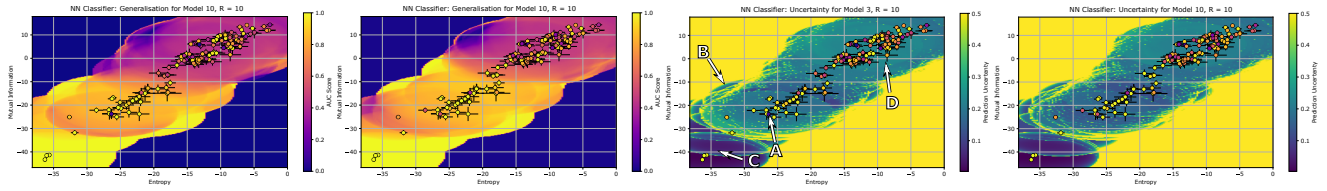
Furthermore, some features that can be seen in the uncertainty estimate are discussed. The areas that are discussed in the rest of this section are marked in figure 6c (A-D).
First of all, the classifier outputs a high uncertainty in areas where there are no data points, which is by design. However, there are some regions (B,C) where the classifier is overconfident. Because the smallest error here is assigned to predictions that are based on a few values. Additionally these values are at the edge of the space that is taken into account for prediction. Further investigation needs to be done here. To clearly judge if this is an issue or not, the probability that a data set actually is indexed by a FEMI-index in that region needs to be quantified.

In the regions where there are points, the uncertainty shows the desired behavior, that, in the region where the AUC-score is not directly associated with the FEMI-index (D, upper right). The error is higher than in the regions where there is an association. Even though, for our liking, this error could be higher, maybe even as large as 0.5.

Another feature worth mentioning is the huge influence of that one deviating point (A) on the prediction. However, this influence is reflected in the uncertainty.

6 CONCLUSION

In this paper, it is described how information theoretical concepts can be used to characterize the properties of a time series data set. An association between this measure and the AUC-Score of a model that was trained on that data is visible for the trained models.
To investigate this association and quantify it, a nearest neighbor classifier was created, featuring an uncertainty classification based on Gaussian error propagation.
A classifier that always outputs the mean of the measured AUC-scores was identified as the baseline. Surpassing the performance of this classifier is only possible if further information from the index is used.
The nearest neighbor classifier performed within the margin of error of a classifier that outputs the average of the measured AUC-scores. This means that no clear mathematical evidence for the association that can be seen in the visualization of the index could be shown.
To really grasp how the association between FEMI-index and classifier performance comes to be, further investigation should be conducted. Up to this point, we just reported empirical results. A theoretical background that explains the reported results is needed. We personally think that the concepts discussed in this paper are worth researching. A way to mathematically estimate the difficulty



(a) generalization for model 3 **(b)** generalization for model 10 **(c)** uncertainty for model 3 **(d)** uncertainty for model 10

Figure 6: The left two plots (a and b) show how the nearest neighbor classifier generalizes for the models 3 (a) and 10 (b). The right two plots (c and d) show the uncertainty the classifier estimates for its prediction. Additionally in figure c, there are some annotated areas. The annotations are used in the discussion of these plots in the text.

associated the anomaly detection task for a dataset would be a huge benefit to the field. The workflow proposed (Build NN-classifier based on the index, compare against the average classifiers MSE) in this paper is a methodology to evaluate and compare mathematical measures that are meant to classify data sets.

7 FUTURE WORK

The future work section is split into two subsections. In one subsection the extensions and additions to presented experiments are pointed out. The other shows potential variations and points to "branch off" from here.

7.1 Improvements to the existing methodology

As discussed above (section 3), there are some points to extend the experiments done here.

- The results shown here are all empirical. A theoretical background that explains the findings is needed.
- To get a more detailed picture of the FEMI-index, more data sets (At the moment most of the data for the tesing originates from the UCR time series classification data set) need to be added to the existing experiments.
- For the convolutional and the feed-forward networks, the tendencies visible in the FEMI-index looked roughly the same (chaos in the upper right and mostly good datasets in the lower left). It would be interesting to compute the FEMI-index on more different models to see if this general behavior changes with the model type.
- During Testing, there where outiliers. Points that are in a region, where the FEMI-Index suggests, that the anomaly detection algorithm would perform well on them, that had lower AUC-Score than the points surrounding them. It would be interesting to investigate, which properties make these points an outilier.
- As suggested by Clara Hoffmann[3], there are known transformations that can be done to the data, which don't change the shannon entropy. It would be interesting to see how the different data with same FEMI-index would look like.
- Christian Schlauch[2] proposed a different interpretation for the FEMI-index: We interpreted the relation of the AUC-values with the index as classifier dependent. Maybe this is only partially true. It could as well be that the FEMI-index is a theoretical upper bound for the performance of any classifier.

7.2 Variants of the research presented here

There are some promising aspects of the FEMI-index and the opportunities that might arise when it is used as a basis for selecting data sets for benchmarking or for pre-training performance estimation. But, as we stated in the former sections, there still is a lot to desire. Mainly a mathematical way of quantifying the association between classifier performance and FEMI-index. During our investigation on this topic, we identified some potential variants of the algorithm that might be worth exploring.

- There might be other information theoretical measures to explore to assign an index to a data set. E. g. Different entropy measures. Another parameter that might be worth including in an index is the estimates signal to noise ratio.
- The uncertainties in this paper were huge compared to the numeric values. Maybe there is a way to find an index for the classifier that is less prone to error.

There are other, less error-prone, more precise classifiers on the basis of the existing FEMI-index. At the moment, we chose a relatively simple classifier for the sake of interpret-ability.

- There are different algorithms that can be used for classifications, like e. g. forests or neural networks, that are less easy to interpret but hopefully achieve better classification.
- There are variants for the classifier at hand. e. g. trying different weight functions, making the radius dependent on the position in FEMI-parameter space, or choosing a distance metric for creating the circle that weights mutual information differently compared to entropy.

ACKNOWLEDGMENTS

This work was supported by the Observatory on Artificial Intelligence in Work and Society as part of the Policy Lab Digital, Work & Society of the Federal Ministry for Labour and Social Affairs (BMAS).

This work was supported by the Research Center Trustworthy Data Science and Security, an institution of the University Alliance Ruhr.

REFERENCES

[1] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. 2019. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*. Springer, 622–637.

[2] Christian Schlauch (christian.schlauch@student.hu berlin.de). 2023. personal communication.

[3] Clara Hoffmann (clara2.hoffmann@tu dortmund.de). 2023. personal communication.

[4] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. 2019. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica* 6, 6 (2019), 1293–1305.

[5] Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh. 2018. ECG Heartbeat Classification: A Deep Transferable Representation. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE. https://doi.org/10.1109/ichi.2018.00092

[6] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

[7] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Physical review E* 69, 6 (2004), 066138.

[8] Kwei-Herng Lai, Daochen Zha, Junjie Xu, Yue Zhao, Guanchu Wang, and Xia Hu. 2021. Revisiting time series outlier detection: Definitions and benchmarks. In *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 1)*.

[9] Pankaj Malhotra, Vishnu TV, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2017. TimeNet: Pre-trained deep recurrent neural network for time series classification. arXiv:1706.08838 [cs.LG]

[10] Iván Marín-Franch, Martín Sanz-Sabater, and David H. Foster. 2022. Application of offset estimator of differential entropy and mutual information with multivariate data. *Experimental Results* 3 (2022). https://doi.org/10.1017/exp.2022.14

[11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red Hook, NY, USA.

[12] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. 2022. Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment* 15, 9 (2022), 1779–1797.

[13] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2828–2837.

[14] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. 2022. TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data. arXiv:2201.07284 [cs.LG]

[15] Kush R Varshney. 2021. Trustworthy machine learning. *Chappaqua, NY* (2021).

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[17] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. 2022. Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy. arXiv:2110.02642 [cs.LG]

[18] Ilia Zaitsev. [n. d.]. [PyTorch] Deep Time Series Classification. https://www.kaggle.com/code/purplejester/pytorch-deep-time-series-classification. Accessed: 2010-09-30.

[19] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*.

A APPENDIX

A.1 MSE for the classifier for all models

The following tables show the classifier MSE archived for all models:

A.2 Example plots for the Transformer- and RNN-Models

The following part of the appendix has more detailed information on the models that were omitted from the discussion of the result in the main part of this work due to technical problems. The specific model parameters can be seen in table 4.

LSTM / GRU (4-7). These models consist of a GRU/LSTM that processes the input. The information is encoded in the cell states by one recurrent layer stack and then passed to another that should rebuild the input. Additionally, there is the option to flip the cell states before they are passed to the second layer stack and flip the output so that the network rebuilds the input from last to first like it's described by Malhorta et al. [9].

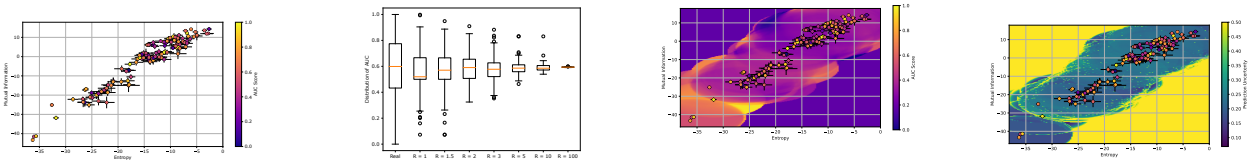
Attention Based Model (13-14). This model is a wrapper for the `pytorch` implementation of the "attention is all you need" transformer [16] (This is a different approach than what is done in the state of the art transformer based anomaly detection algorithms, where the anomalies are detected by evaluating the attention mechanism instead of the reproduction. [14][17]). The idea behind this model is as follows: Both time series and speech can contain complex context-sensitive information. So the idea arises that the attention mechanism, which enables the transformer to process this information in speech, also brings benefits when processing time series.
In the wrapper used here, instead of word embedding, the time series is either fed directly (piecewise) into the transformer, or a version of the time series that is preprocessed by a feed-forward neural network is passed. The output of the transformer is expanded back to the dimensionality of the input by another feed-forward network.

Table 3: This table shows the MSE of our two baseline classifiers and the nearest neighbor classifier for different values of R for the polar (P:) and component (C:) FEMI-index. All uncertainties are obtained by error propagation.

Error/Model ID	0		1		2		3		4	
Random C. MSE	0.212 ± 0.022		0.218 ± 0.017		0.214 ± 0.016		0.208 ± 0.018		0.144 ± 0.016	
Avg. C. MSE	**0.046 ± 0.031**		**0.052 ± 0.021**		**0.041 ± 0.025**		**0.055 ± 0.020**		**0.052 ± 0.013**	
NN-C. MSE	P:	C:	P:	C:	P:	C:	P:	C:	P:	C:
R = 1	**0.081 ± 0.279**	**0.088 ± 0.297**	**0.097 ± 0.301**	**0.103 ± 0.315**	**0.088 ± 0.289**	**0.096 ± 0.305**	**0.091 ± 0.288**	**0.100 ± 0.307**	**0.077 ± 0.258**	**0.067 ± 0.236**
R = 1.5	**0.074 ± 0.252**	**0.083 ± 0.286**	**0.085 ± 0.268**	**0.090 ± 0.289**	**0.071 ± 0.251**	**0.079 ± 0.268**	**0.087 ± 0.273**	**0.092 ± 0.283**	**0.072 ± 0.237**	**0.070 ± 0.226**
R = 2	**0.055 ± 0.209**	**0.069 ± 0.236**	**0.062 ± 0.221**	**0.079 ± 0.251**	**0.049 ± 0.194**	**0.065 ± 0.231**	**0.063 ± 0.220**	**0.077 ± 0.240**	**0.063 ± 0.204**	**0.066 ± 0.188**
R = 3	**0.046 ± 0.165**	**0.060 ± 0.202**	**0.054 ± 0.178**	**0.067 ± 0.210**	**0.041 ± 0.150**	**0.052 ± 0.183**	**0.055 ± 0.179**	**0.070 ± 0.200**	**0.056 ± 0.159**	**0.068 ± 0.186**
R = 5	**0.044 ± 0.113**	**0.047 ± 0.133**	**0.052 ± 0.123**	**0.056 ± 0.146**	**0.039 ± 0.107**	**0.042 ± 0.121**	**0.052 ± 0.122**	**0.058 ± 0.150**	**0.056 ± 0.121**	**0.056 ± 0.125**
R = 10	**0.040 ± 0.095**	**0.041 ± 0.097**	**0.048 ± 0.104**	**0.049 ± 0.110**	**0.037 ± 0.088**	**0.036 ± 0.088**	**0.048 ± 0.104**	**0.050 ± 0.111**	**0.054 ± 0.113**	**0.053 ± 0.113**
R = 100	**0.046 ± 0.108**	**0.046 ± 0.108**	**0.053 ± 0.115**	**0.053 ± 0.115**	**0.041 ± 0.095**	**0.041 ± 0.095**	**0.055 ± 0.119**	**0.055 ± 0.120**	**0.053 ± 0.112**	**0.053 ± 0.112**
Error/Model ID	5		6		7		8		9	
Random C. MSE	0.139 ± 0.012		0.168 ± 0.015		0.176 ± 0.013		0.221 ± 0.010		0.225 ± 0.008	
Avg. C. MSE	**0.047 ± 0.010**		**0.064 ± 0.020**		**0.056 ± 0.017**		**0.048 ± 0.028**		**0.049 ± 0.033**	
NN-C. MSE	P:	C:	P:	C:	P:	C:	P:	C:	P:	C:
R = 1	**0.068 ± 0.248**	**0.063 ± 0.248**	**0.099 ± 0.293**	**0.084 ± 0.275**	**0.093 ± 0.291**	**0.082 ± 0.283**	**0.085 ± 0.276**	**0.099 ± 0.308**	**0.090 ± 0.280**	**0.116 ± 0.321**
R = 1.5	**0.065 ± 0.224**	**0.060 ± 0.214**	**0.093 ± 0.267**	**0.085 ± 0.257**	**0.089 ± 0.270**	**0.083 ± 0.261**	**0.078 ± 0.258**	**0.082 ± 0.263**	**0.085 ± 0.265**	**0.098 ± 0.286**
R = 2	**0.058 ± 0.201**	**0.052 ± 0.171**	**0.074 ± 0.232**	**0.079 ± 0.223**	**0.065 ± 0.224**	**0.072 ± 0.223**	**0.050 ± 0.197**	**0.067 ± 0.217**	**0.054 ± 0.204**	**0.079 ± 0.251**
R = 3	**0.052 ± 0.154**	**0.061 ± 0.182**	**0.068 ± 0.187**	**0.074 ± 0.201**	**0.059 ± 0.182**	**0.064 ± 0.195**	**0.042 ± 0.160**	**0.052 ± 0.183**	**0.045 ± 0.170**	**0.066 ± 0.205**
R = 5	**0.052 ± 0.123**	**0.052 ± 0.116**	**0.071 ± 0.154**	**0.070 ± 0.153**	**0.060 ± 0.128**	**0.061 ± 0.147**	**0.044 ± 0.113**	**0.045 ± 0.129**	**0.046 ± 0.116**	**0.050 ± 0.133**
R = 10	**0.047 ± 0.102**	**0.046 ± 0.100**	**0.067 ± 0.136**	**0.067 ± 0.139**	**0.055 ± 0.113**	**0.055 ± 0.117**	**0.044 ± 0.099**	**0.042 ± 0.096**	**0.045 ± 0.100**	**0.045 ± 0.106**
R = 100	**0.048 ± 0.100**	**0.048 ± 0.100**	**0.065 ± 0.138**	**0.065 ± 0.138**	**0.057 ± 0.120**	**0.057 ± 0.120**	**0.049 ± 0.107**	**0.049 ± 0.108**	**0.049 ± 0.112**	**0.049 ± 0.113**
Error/Model ID	10		11		12		13		14	
Random C. MSE	0.212 ± 0.011		0.224 ± 0.008		0.214 ± 0.011		0.181 ± 0.009		0.185 ± 0.011	
Avg. C. MSE	**0.048 ± 0.026**		**0.054 ± 0.031**		**0.050 ± 0.034**		**0.062 ± 0.018**		**0.060 ± 0.019**	
NN-C. MSE	P:	C:	P:	C:	P:	C:	P:	C:	P:	C:
R = 1	**0.088 ± 0.283**	**0.106 ± 0.316**	**0.100 ± 0.296**	**0.123 ± 0.328**	**0.098 ± 0.295**	**0.113 ± 0.317**	**0.096 ± 0.301**	**0.101 ± 0.301**	**0.096 ± 0.304**	**0.103 ± 0.307**
R = 1.5	**0.071 ± 0.248**	**0.083 ± 0.265**	**0.096 ± 0.276**	**0.108 ± 0.294**	**0.087 ± 0.267**	**0.092 ± 0.272**	**0.093 ± 0.271**	**0.101 ± 0.271**	**0.093 ± 0.279**	**0.104 ± 0.277**
R = 2	**0.052 ± 0.195**	**0.067 ± 0.225**	**0.062 ± 0.216**	**0.090 ± 0.255**	**0.056 ± 0.208**	**0.073 ± 0.233**	**0.075 ± 0.241**	**0.098 ± 0.251**	**0.075 ± 0.247**	**0.098 ± 0.247**
R = 3	**0.043 ± 0.153**	**0.050 ± 0.182**	**0.053 ± 0.180**	**0.072 ± 0.197**	**0.048 ± 0.168**	**0.067 ± 0.197**	**0.065 ± 0.198**	**0.081 ± 0.217**	**0.064 ± 0.202**	**0.077 ± 0.220**
R = 5	**0.042 ± 0.116**	**0.045 ± 0.121**	**0.053 ± 0.128**	**0.059 ± 0.145**	**0.047 ± 0.115**	**0.052 ± 0.138**	**0.064 ± 0.152**	**0.068 ± 0.157**	**0.061 ± 0.141**	**0.065 ± 0.151**
R = 10	**0.041 ± 0.095**	**0.042 ± 0.099**	**0.052 ± 0.112**	**0.051 ± 0.114**	**0.044 ± 0.095**	**0.045 ± 0.101**	**0.063 ± 0.134**	**0.060 ± 0.133**	**0.060 ± 0.127**	**0.059 ± 0.132**
R = 100	**0.048 ± 0.109**	**0.048 ± 0.109**	**0.055 ± 0.121**	**0.055 ± 0.122**	**0.050 ± 0.115**	**0.050 ± 0.115**	**0.063 ± 0.131**	**0.063 ± 0.131**	**0.061 ± 0.126**	**0.061 ± 0.126**

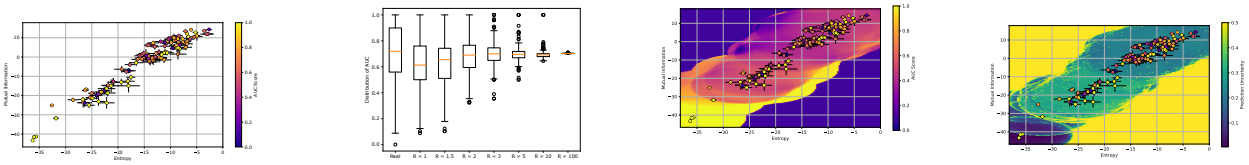
Table 4: This table lists the details of the models which were excluded from the discussion of the results due to errors in the implementation (transformer) and stability problems while training (LSTM / GRU).

Model ID	Model Type	Comment
4	LSTM	performes latent flip
5	GRU	performes latent flip
6	LSTM	no latent flip
7	GRU	no latent flip
13	Attention-based Model	Feed-forward Autoencoder for embedding
14	Attention-based Model	no embedding



(a) FEMI-index for model 4. **(b)** Box plot for model 4. **(c)** Generalisation of the NN-Classifier for model 4 (R = 10). **(d)** Uncertainty for model 4 (R = 10).

Figure 7: These figures show the results discussed for the not properly trained model 4.



(a) FEMI-index for model 14. **(b)** Box plot for model 14. **(c)** Generalisation of the NN-Classifier for model 14 (R = 10). **(d)** Uncertainty for model 14 (R = 10).

Figure 8: These figures show the results discussed for the malfunctioning model 14.