Information Modelling and Knowledge Bases XXXV
M. Tropmann-Frick et al. (Eds.)

doi:10.3233/FAIA231149

Data-Based Condition Monitoring and Disturbance Classification in Actively Controlled Laser Oscillators

Arne GRÜNHAGEN [a,b,c,1], Annika EICHLER [b,c] Marina TROPMANN-FRICK [a] and Görschwin FEY [b]

[a] *Hamburg University of Applied Sciences, HAW, Germany*
[b] *Hamburg University of Technology, TUHH, Germany*
[c] *Deutsches Elektronen-Synchrotron DESY, Germany*

Abstract. The successful operation of the laser-based synchronization system of the European X-Ray Free Electron Laser relies on the precise functionality of numerous dynamic systems operating within closed loops with controllers. In this paper, we present how data-based machine learning methods can detect and classify disturbances to such dynamic systems based on the controller output signal. We present 4 feature extraction methods based on statistics in the time domain, statistics in the frequency domain, characteristics of spectral peaks, and the autoencoder latent space representation of the frequency domain. These feature extraction methods require no system knowledge and can easily be transferred to other dynamic systems. We combine feature extraction, fault detection, and fault classification into a comprehensive and fully automated condition monitoring pipeline. For that, we systematically compare the performance of 19 state-of-the-art fault detection and 4 classification algorithms to decide which combination of feature extraction and fault detection or classification algorithm is most appropriate to model the condition of an actively controlled phase-locked laser oscillator. Our experimental evaluation shows the effectiveness of clustering algorithms, showcasing their strong suitability in detecting perturbed system conditions. Furthermore, in our evaluation, the support vector machine proves to be the most suitable for classifying the various disturbances.

Keywords. Fault detection, Fault classification, Feature Extraction, Autoencoder

1. Introduction and Motivation

The European X-ray Free-Electron Laser (EuXFEL) [1] is a large-scale linear particle accelerator located in Hamburg, Germany. A 1.3 GHz Radio Frequency (RF) Main Oscillator (MO) is used to synchronize various components of the accelerator by distributing the RF signal as a timing reference. Since this electrical distribution via coaxial cables is heavily influenced by the environment (e.g. humidity, temperature, electromagnetic fields), an optical synchronization system is installed that is less vulnerable to these en-

[1] We acknowledge the support by DASHH (Data Science in Hamburg - HELMHOLTZ Graduate School for the Structure of Matter) with the Grant-No. HIDSS-0002.

vironmental condition changes [2]. This optical synchronization system provides ultra-stable reference timing information to the accelerator components and the experimental setups with an integrated timing jitter in the range of a few femtoseconds. The main component of this optical synchronization system is a mode-locked pulsed laser oscillator that is phase-locked to the MO delivering an ultra-stable optical reference used to locally resynchronize RF sources, to lock optical laser systems, and to diagnose the arrival time of the electron beam along various locations for fast beam based feedbacks.

Not only does the laser not produce a completely noise-free signal, but the emitted signal is also influenced by environmental disturbances (i.e., electrical, acoustical, mechanical, and optical) resulting in amplitude and phase fluctuations. To synchronize the laser oscillator to the MO, the relative phase error between a harmonic of the laser pulse repetition rate and the MO reference is determined and fed to a Proportional-Integral (PI) controller in a feedback loop. This controller acts on the laser oscillator cavity length to lock the laser oscillator repetition rate to the 1.3 GHz MO frequency with a loop bandwidth in the order of 1 kHz to 10 kHz [3]. Since the controller compensates for disturbances, the controller output signal is an ideal data source to detect potential disturbances that increase the integrated timing jitter and therefore decrease the synchronization performance.

The aim of this work is to detect and classify changes in the controller output signal which may indicate disturbances of the laser itself, disturbances in the internal detection chain, disturbances in the MO reference or environmental disturbances. This goal is achieved by realizing the fault analysis pipeline depicted in Figure 1. In the data preparation step, we extract the power spectral density (PSD) from the controller output signal using Welch's method [4] such that the fault analysis can be performed in the time and frequency domain. In the feature engineering step, we implemented three different methods to extract meaningful features to fit several fault detection models. The steps will be explained in detail in Sections 3 and 4.

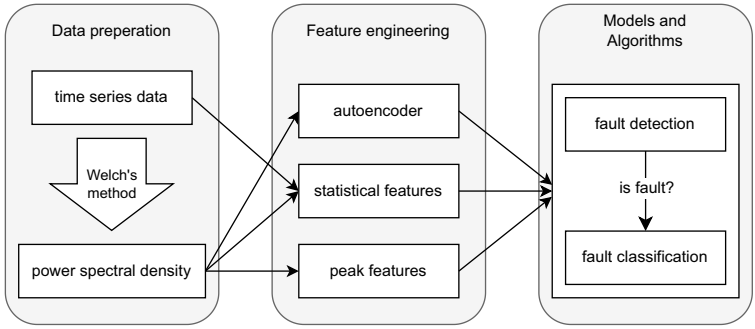

Figure 1. Fault Analysis Pipeline

In the following, we summarize related work in Section 2. Then we describe the data preparation and feature engineering steps in Section 3. Section 4 summarizes the methods selected for fault detection and fault classification and Section 5 gives a detailed overview on the experimental validation of the proposed fault analysis pipeline. We conclude this work highlighting specific findings and providing an outlook on future work in Section 6.

2. Related Work

Despite extensive literature about fault detection and anomaly detection in the area of manufacturing systems [5,6,7,8,9] only a few publications address fault detection and fault classification of dynamic systems in closed-loop control.

The authors of [10] use linear transfer functions to represent the actively controlled system under review and its controller. These models build the core of their fault diagnosis since they evaluate the discrepancy between the physical system output and the model output and the discrepancy between the physical controller output and the model output. In [11], the authors address control-loop data from a real system. They implement different fault detection mechanisms for different fault types, namely an oscillation detection based on an autocorrelation function, the detection of sluggish-tuned loops using the so-called idle index, quantization detection, and a saturation detection method. Both approaches require a deep system understanding for analyzing faulty system behavior.

In contrast to existing fault analysis targeting dynamic systems in closed loop control, we aim to develop a fault analysis pipeline that is purely based on historical data. Our fault diagnosis uses a combination of automatic feature extraction and data driven machine learning techniques. Feature engineering is addressed in different industrial sectors. The authors of [12,13,14] each extract different basic statistics in the time domain, like the mean, the maximum, the minimum, the root mean square, or the entropy. In [15,16], the authors analyze vibration signals in the frequency domain and decide on the system's health condition based on the values of domain-relevant frequency components. In [17] the authors combine both, statistics in the time domain and statistics in the frequency domain as features for fault analysis. In our research, we extract features from the controller output signal in the time and frequency domains and fit state-of-the-art data driven fault detection and fault classification algorithms to these features.

3. Data Engineering

In this section, we describe what kind of data is used and how the data is processed to build meaningful models that can describe the condition of laser oscillators.

Figure 2 shows a simplified version of the laser oscillator control loop. The input $e(t)$ to the PI controller is the difference between the reference signal $r(t)$, which in our case is the phase of the reference signal provided by the electrical timing information coming from the MO, and the measured phase $y_m(t)$ of the signal generated by the laser oscillator, affected by different disturbances $d(t)$. The phase $y(t)$ of the laser oscillator output signal $o(t)$ is determined by the phase detection. The output $u(t)$ of the PI controller feeding into the laser oscillator is a voltage that affects the cavity length of the laser oscillator and therby adjusting the phase of the laser signal. This outgoing signal, also called feedback signal, contains information about disturbances that the PI controller is processing and is therefore a valuable source of information for fault detection.

3.1. Time and Frequency Domain

The controller's output signal contains values in the range from 0 to 1 and is measured with a sampling rate of 0.32 MHz. To check what kind of disturbances affect the system, the operators of the optical synchronization system mainly study the PSD.

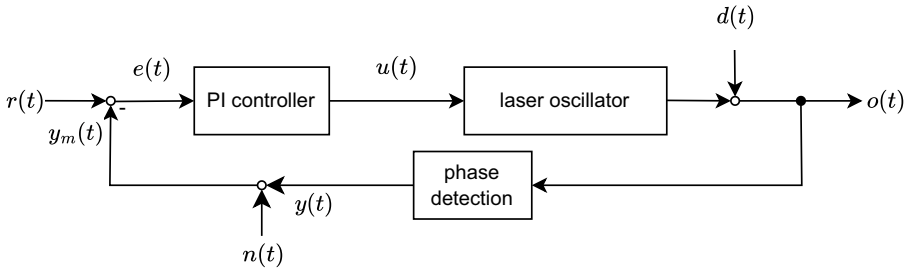

Figure 2. Overview of the Laser Oscillator Control Scheme

Figure 3 shows an examples of a feedback signal in time domain and the respective PSD in the frequency domain during healthy operation. The time series signal is an oscillating signal containing the adjustments to the cavity length of the laser oscillator. Due to the oscillating nature of the feedback signal in the time domain, single data points cannot reflect the entire state of the system and therefore it is mandatory to look at a series of data points. For our calculations, each series contains 30 000 datapoints, which is equivalent to 0.1 s.

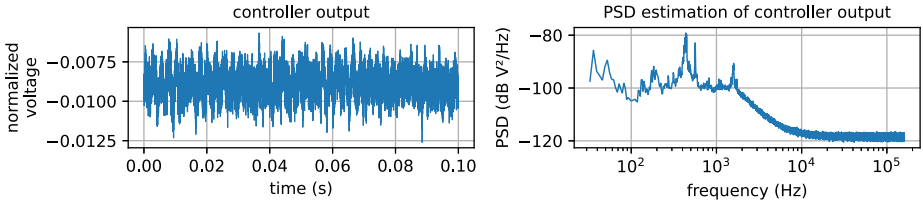

Figure 3. Example for time series signal and PSD during normal operation

We calculate the PSDs using Welch's method [4]. Welch's method divides the time series data into overlapping segments, computes a modified periodogram for each segment, and averages the periodograms to the resulting PSD. Our PSD calculation uses Hanning windows containing 10 000 data points with an overlap of 5000 datapoints. As a result, each PSD consists of 5000 datapoints. The shape of the PSD and its peaks at certain frequencies are characteristic of the current state of the system. For example, the increased power at 400 Hz comes from a mechanical disturbance of the laser oscillator and the peak at 60 000 Hz originates from the resonance frequency of the laser internal piezo actuator (see Figure 3).

In either case, considering the time-domain signals or the PSD in frequency domain, we work with a series of data points, also called frames. Depending on the frame size, the fault analysis algorithms may have to work with high dimensional data, which can lead to poor prediction performance. For this reason, we use several feature engineering techniques to reduce the dimensionality of the input data. In the following, we describe three feature engineering techniques applied to the controller data.

3.2. Statistical Feature Extraction

We use the *tsfresh* Python package [18] to calculate a set of statistics from the data frames. The selected statistics correspond to the most frequently used statistics in the

field of feature extraction for fault analysis [19]. Table 1 gives an overview of the extracted statistics, a short description, and, if applicable, the corresponding parameter choices. These statistical features are extracted from the time frames and from the PSDs. In both cases, the resulting dataset contains 34 values for the time-domain frame or PSD, respectively.

Table 1. Summary of Extracted Statistics from Data Frame `X` Containing `n` Elements

Statistic	Description	Parameter values
maximum	$\max(\mathtt{X})$	-
minimum	$\min(\mathtt{X})$	-
mean	$\mu = \frac{\sum_{i=1}^{n} X_i}{n}$	-
absolute energy	$\frac{\sum_{i=1}^{n} X_i^2}{n}$	-
standard deviation	$std = \sqrt{\frac{\sum_{i=1}^{n}(X_i-\mu)^2}{n}}$	-
variation coefficient (cv)	$cv = \frac{std}{\mu}$	-
variance (σ)	$\sigma = (std)^2$	-
root mean square (rms)	$\text{rms} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} X_i^2}$	-
skewness (sk)	$sk = \frac{\sum_{i=1}^{n}(X_i-\mu)^3}{n \cdot S^3}$	-
kurtosis ($kurt$)	$kurt = \frac{\sum_{i=1}^{n}(X_i-\mu)^4}{n \cdot S^4} - 3$	-
autocorrelation for lags	$R(l) = \frac{1}{(n-l)\sigma^2}\sum_{t=1}^{n-l}(x_t-\mu)(x_{t+l}-\mu)$	$l \in [0, 1, ..., 9]$
quantile	The quantile is the value that is greater than the `q`-th proportion of the values in `X`.	`q` $\in$ [0.1, 0.2, ..., 1]
linear trend attributes	pvalue, rvalue, intercept, slope, stderr of the linear regression from `X`	-

3.3. Peak Feature Extraction

The peak feature extraction addresses data in the frequency domain. Peaks within a power spectrum are special characteristics since they show how much the PI controller was correcting and at which frequency. This behavior provides insights about possible disturbances. We define a potential peak as every point in a series of data points which has a value higher than both of its neighboring data points. We filter out the irrelevant peaks, i.e. peaks due to noise, by specifying that a peak should have a minimum prominence of 45 dbm and a minimum value of -105 dbm. The prominence of a peak is defined as the vertical distance to the highest valley. The minimum height and the minimum prominence are specific to the controller output signal. Based on this peak detection we implemented three feature extraction algorithms that identify peak related characteristics.

3.3.1. Number of Peaks per Area

We divided the frequency range in smaller regions, each covering 5000 Hz, and count the number of peaks. With a maximum frequency of 160 000 Hz we have a total number of 32 features.

3.3.2. Characteristics of the Most Prominent Peaks

We identified the five most prominent peaks, from which we extract the prominence, the height, the width, and the frequency. While the prominence, the height, and the width are numerical values, the frequency is a categorical value because a higher frequency does not imply a worse or better system condition. Therefore, we again divided the frequency range into regions of 5000 Hz, and for each region we count the number of prominent peaks.

3.3.3. Peak Healthyness

This feature extraction method gives each extracted peak following our set of constraints a score between 0 and 1 that determines whether the peak belongs to a healthy or unhealthy operation. For that we acquired controller output data during healthy operation and extracted all peaks following our criteria from the PSDs. Based on these peaks we assigned each frequency f a healthyness score $healthyness(f) = \frac{\#\,peaks\,at\,f}{\#\,observed\,PSDs}$. The resulting distribution of healthy peaks is depicted in Figure 4.

In the feature extraction step, we identify the ten most prominent peaks and for each peak we take the healthyness score from the previously determined distribution as a feature.

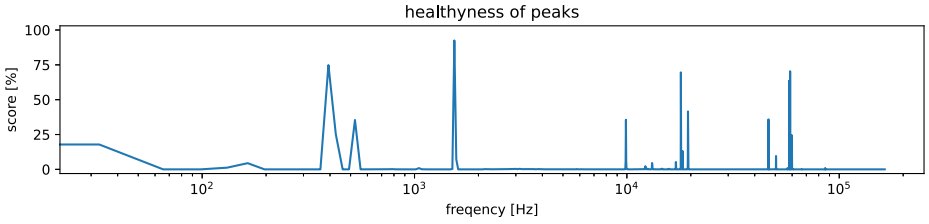

Figure 4. Probability Distribution of Frequencies Having a Healthy Peak

3.4. Autoencoder Latent Space

We use a feedforward AutoEncoder (AE) [20] trained on PSDs from healthy and disturbed operations. Using the AE's encoder, we transform a complete PSD into the AE latent space vector, which is used as a feature vector for fault detection methods. The basic structure of the AE is shown in Table 2. The AE consists only of fully connected layers and each layer, except for the output layer, is followed by the `leakyRELU` activation function.

Table 2. Overview Autoencoder

Layer	input	1. encoding	2. encoding	latent space	1. decoding	2. decoding	output
Dimension	5000	500	100	10	100	500	5000

4. Selected Models for Fault Detection and Fault Classification

The fault analysis pipeline depicted in Figure 5 is composed of two integral components: a semi-supervised fault detection algorithm based on healthy data and a supervised fault classification algorithm fitted on predefined classes. These algorithms are fed with input features extracted from 0.1 s snippets of the system's behavior. The fault detection performs an assessment of the system's condition, distinguishing between normal operation and states of malfunction or disturbance.

If the fault detection classifies a system state as faulty based on the input features, the same features are used as input for fault classification to distinguish between different system conditions. The supervised fault classification algorithms assigns a class score to each of the predefined predefined system conditions. This class score reflects the likelihood that the given 0.1 s snippet corresponds to a specific fault class. Through the synergistic operation of fault detection and fault classification, the pipeline enables the determination of the system's current health condition.

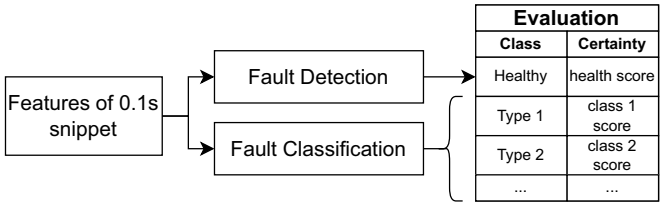

Figure 5. Fault Detection and Fault Classification

4.1. Fault Detection Algorithms

The fault detection algorithms are used to evaluate the features x extracted from a 0.1 s snippet of measurement data. For a vector x, the fault detector calculates a health score H, which quantifies the operational health of a system. This score ranges from 0 to 1, where 0 represents a highly faulty or disturbed state and 1 represents a completely healthy state.

4.1.1. Clustering Algorithms

Clustering algorithms aim to group data samples into classes with similar elements. Clustering requires the concept of a metric, which may differ from algorithm to algorithm [21]. For the purpose of fault detection, we assume that similar data samples belong to the same class. We use the following clustering algorithms:

- Balanced iterative reducing and clustering using hierarchies (BIRCH) [22]
- Clustering based local outlier factor (CBLOF) [23]
- Gaussian mixture model (GMM) [24]
- K-means clustering [25]

4.1.2. Outlier Detection Algorithms

Outlier detection algorithms aim to identify rare items or events that differ significantly from the rest of the dataset [26]. Assuming that faulty data samples can be classified as outliers compared to healthy data samples, we use the following outlier detection algorithms:

- Angle-based outlier detection (ABOD) [27]
- Connectivity-based outlier detection (COF) [28]
- Copula-based outlier detector (COPOD) [29]
- Empirical cumulative distribution outlier detection (ECOD) [30]
- Isolation-based outlier detection (IOF) [31]
- Kernel density estimation (KDE) [32]
- Kernel principal component analysis (KPCA) [33]
- K-nearest neighbor detection (KNN) [34]
- Linear model deviation-based outlier detection (LMDD) [35]
- Local outlier factor (LOF) [36]
- Minimum covariance determinant (MCD) [37]
- One-class support vector machine (OCSVM) [38]
- Principal component analysis (PCA) [39]
- Sampling [40]
- Stochastic outlier selection (SOS) [41]

In addition to the algorithms fitted on the feature dataset, we trained an AE with the structure shown in Table 2 on PSDs belonging to healthy system operation. Therefore, the AE only learns to reconstruct PSDs belonging to a healthy operation. The AE fault detector uses a threshold on the reconstruction loss, which is realized using the mean squared error (MSE) between the input PSD and the reconstructed PSD at the output layer. The fault detection is based on the assumption that PSDs belonging to healthy system operation have a low MSE, while PSDs belonging to poor system conditions have a high MSE.

4.2. Fault Classification

Fault classification is the process of categorizing a feature vector x into specific fault classes using algorithms that are built using labeled training data. For each fault class $C_1, C_2, ..., C_n$, the fault classifier assigns a feature vector x to the fault class C_i with the maximum likelihood $P(C_i|x)$:

$$C = \arg\max_i P(C_i|x)$$

The following supervised learning algorithms use feature vectors as input:

- Decision Tree (DT) [42]
- k-Nearest-Neighbor classifier (kNN) [43]
- Random Forest (RF) [44]
- Support Vector Machine (SVM) [45]

5. Experimental Evaluation

The experiments were performed using the Python libraries tsfresh [18], PyOD [46], and Scikit-learn [47]. The runtimes were measured on a Windows 11 operating system running Python 3.9 with a processor Intel(R) Core(TM) i7-1185G7 @ 3.00 GHz and 16 GB of RAM.

5.1. Dataset Summary

To evaluate the feature extraction techniques in combination with the fault detection and fault classification algorithms we generated disturbances at different frequencies by playing tones of single frequencies. The tones were played through a surface speaker mounted on the optical table directly next to the laser oscillator. Figure 6 shows exemplary the time series domain of the controller output and its corresponding PSD for a 1.0 kHz and a 2.0 kHz excitment. In total, we recorded 30 s of controller data for each excited frequency as summarized in Table 3.

Table 3. Summary of fitting dataset and validation dataset

Condition	Fitting data		Validation data	
	# Frames	**Portion**	**# Frames**	**Portion**
no disturbance	4208	60.49 %	231	7.97 %
0.5 kHz disturbance	305	4.38 %	296	8.97 %
1.0 kHz disturbance	305	4.38 %	296	8.97 %
1.5 kHz disturbance	306	4.4 %	296	10.22 %
2.0 kHz disturbance	305	4.38 %	296	10.22 %
2.5 kHz disturbance	305	4.38 %	296	10.22 %
3.0 kHz disturbance	305	4.38 %	231	10.25 %
3.5 kHz disturbance	306	4.4 %	296	10.22 %
4.0 kHz disturbance	306	4.4 %	296	10.22 %
4.5 kHz disturbance	306	4.4 %	296	10.22 %

For evaluating the combinations of feature extraction and fault analysis we recorded fitting data and validation data under the same environmental conditions (21 °C, 44 % relative humidity). From both, the time frames and the PSDs, we extracted the features as described in Section 3 and normalized the extracted features using Z-normalization [48]. The number of features per data frame depends on the feature extraction method and is shown in Table 4. The peak characteristic feature extraction leads to the highest dimension and the AE latent space feature extractor to the lowest.

Table 4. Numer of Features per Dataframe

Feature Extraction Method	statistics (time)	statistics (PSD)	peak characteristics (PSD)	AE latent space (PSD)
Number of extracted features	34	34	94	10

Figure 7 shows the four fitting data sets of the four different feature extraction methods (AE latent space, statistics in the time domain, statistics from PSDs, and peak fea-

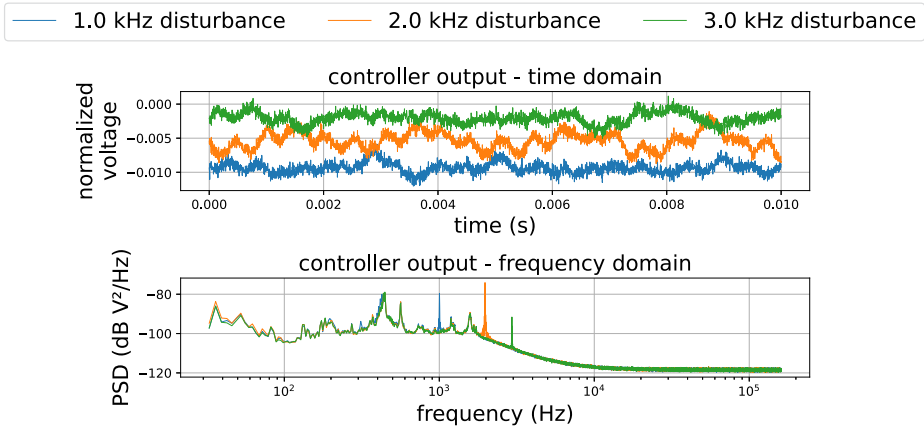

Figure 6. Controller output in time domain and frequency domain at 1.0 kHz, 2.0 kHz, and 3.0 kHz disturbances

tures). To represent the multidimensional feature datasets in a two-dimensional space, we used t-distributed Stochastic Neighbor Embedding (t-SNE) [49]. The visualization of the data is intended to provide a basis for evaluating the algorithms in Section 5.4.

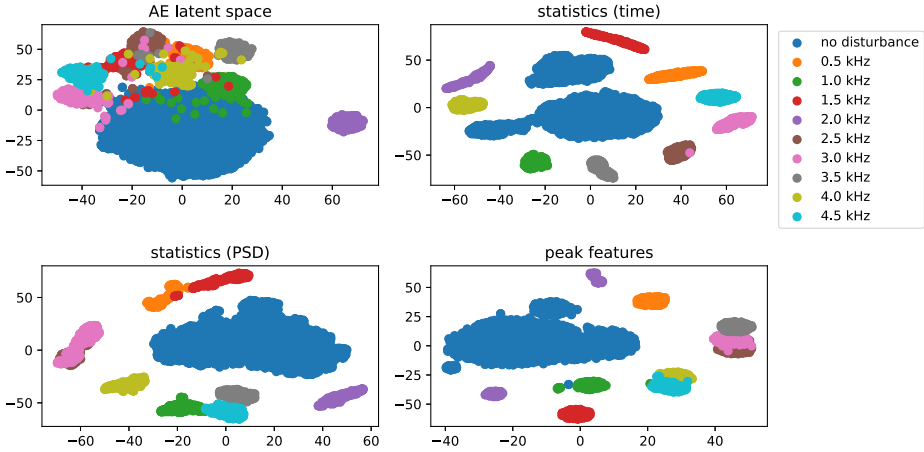

Figure 7. Feature Vizualization by t-SNE

The data points recorded under the same disturbances or no disturbance are clustered for all feature extraction methods. However, the clusters on the AE latent space dataset are significantly closer, sometimes even with an overlap, than the clusters on the other feature datasets. Since we are analyzing fault detection methods in this work, it is noteworthy that there is only an overlap between disturbed data points and undisturbed data points on the AE latent space dataset. In particular, the data from the 1.5 kHz disturbance have a strong overlap with the undisturbed data points. Using statistics from time series, statistics from PSDs, or peak features there are only overlapping clusters between data points of different disturbance types. Furthermore, it is noticeable that the undisturbed datapoints based on time statistics form two separate clusters rather than one cluster.

5.2. Algorithms Parameters

Most of the algorithms selected contain controllable parameters that influence different aspects of the algorithm. A summary of all the parameters used is given in Tables 5,6, and 7.

Table 5. Fault Detection Clustering Algorithms' Parameters

Algorithm	Parameter	Values
BIRCH	threshold	0.2, 0.4, ..., 3.8, 4.0
	branching factor	20, 40, 60, 80, 100
CBLOF	# clusters	2, 3, 4, 5, 6, 7, 8, 9, 10, 15
	alpha	0.5, 0.6, 0.7, 0.8, 0.9
	beta	1.5, 2, 5, 7, 10, 15
GMM	# components	2, 3, 4, 5, 6, 7, 8, 9, 10, 15
K-means	# clusters	2, 3, 4, 5, 6, 7, 8, 9, 10, 15

Table 6. Fault Detection - Outlier Detection Algorithms' Parameters

Algorithm	Parameter	Values
ABOD	# nearest neighbors	5, 10, ..., 95, 100
COF	# nearest neighbors	5, 10, ..., 95, 100
COPOD	-	-
ECOD	-	-
KDE	bandwith	0.2, 0.4, 0.6, ..., 3.8, 4.0
KNN	# nearest neighbors	5, 10, ..., 95, 100
KPCA	# components	1, 2, ..., 10
	kernel	linear, poly, rbf, sigmoid
LMDD	-	-
LOF	# nearest neighbors	5, 10, ..., 95, 100
MCD	-	-
OCSVM	kernel	linear, poly, rbf, sigmoid
PCA	# components	1, 2, ..., 10
Sampling	-	-
SOS	perplexity	5, 10, ..., 95, 100

5.3. Performance Criteria

We repeated the experiment under the same environmental conditions (21 °C, 44 % relative humidity) to verify the quality of the algorithms. The validation data again consists

Table 7. Fault Classification Algorithms' Parameters

Algorithm	Parameter	Values
DT	split criterion	gini, entropy, log-loss
kNN	# neighbors	5, 10, 50, 100, 500
	neighborhood weights	uniform, distance
	algorithm	ball-tree, kd-tree, brute
	distance metric	minkowski, cityblock, euclidean, manhattan
RF	# estimators	10, 50, 100, 200, 500
	split criterion	gini, entropy, log-loss
SVM	regularization parameter	0.25, 0.5, .., 2.0
	kernel	linear, poly, sigmoid, rbf

of 2896 data frames, each covering 0.1 s. A live fault detection or classification of a 0.1 s snippet can have a maximum inference duration of 0.1 s. To evaluate the system state at each point in time, it is necessary that the fault detection and classification algorithms operate at the same or higher speed. We evaluate this criterion by measuring the time it takes for each algorithm to classify the validation data samples and determine the inference speed by dividing the measured duration by the number of frames. Additionally, we measured the time each algorithm needs to be fitted.

To evaluate the feature extraction methods and algorithms qualitatively we are using the area under the receiver operating characteristic (AUROC) [50] as a performance metric. The AUROC score is defined as the area underneath the ROC curve and ranges between 0 and 1, where a score of 1 implies a perfect predictor, an AUROC score of 0 implies that the predictor gives always wrong predictions, and an AUROC score of 0.5 indicates that the predictor makes random guesses. We calculate the AUROC scores of both, the fitting dataset and the validation dataset.

The AUROC metric does not provide information on which of the disturbances are classified correctly and which of them are misclassified. Therefore, we also calculate the classification accuracy $\frac{TruePredictions(condition)}{AllPredictions(condition)}$ for each condition, either a disturbed frequency or undisturbed respectively.

We repeated the process of fitting and evaluation ten times with different random seeds and determined the mean value for each metric. In summary, we determined the mean values of the following metrics for each combination of feature extraction method and fault detection algorithm:

- Fitting duration (fitting dataset)
- Inference duration (validation dataset)
- AUROC score (fitting dataset)
- AUROC score (validation dataset)
- Condition specific accuracies (validation dataset)

5.4. Results

In this section, we describe the results of the algorithms applied to the experimental data. Combining the feature extraction methods and the different parameter choices, we

built 3084 models on the different feature datasets (AE latent space, statistics from time series signals, statistics from PSDs, peak characteristics). We systematically traversed the parameters outlined in Tables 5, 6, and 7 in a nested loop iteration procedure.

The fitting durations of all algorithms related to the feature extraction method and the choice of parameters are depicted in Figure 8. The fitting durations only include the fitting of the algorithms and not the transformation of the recorded data into the features.

All clustering algorithms require very little time to be fitted for all feature extraction methods and all parameter choices. Among the outlier detection algorithms, the LMDD algorithm and ABOD have by far the longest fitting durations. Among the other algorithms, KPCA needs the longest time to be fitted for all feature extraction methods. It is noticeable that KPCA using the AE latent space features takes more than 100 seconds longer to be fitted than the other feature extraction methods.

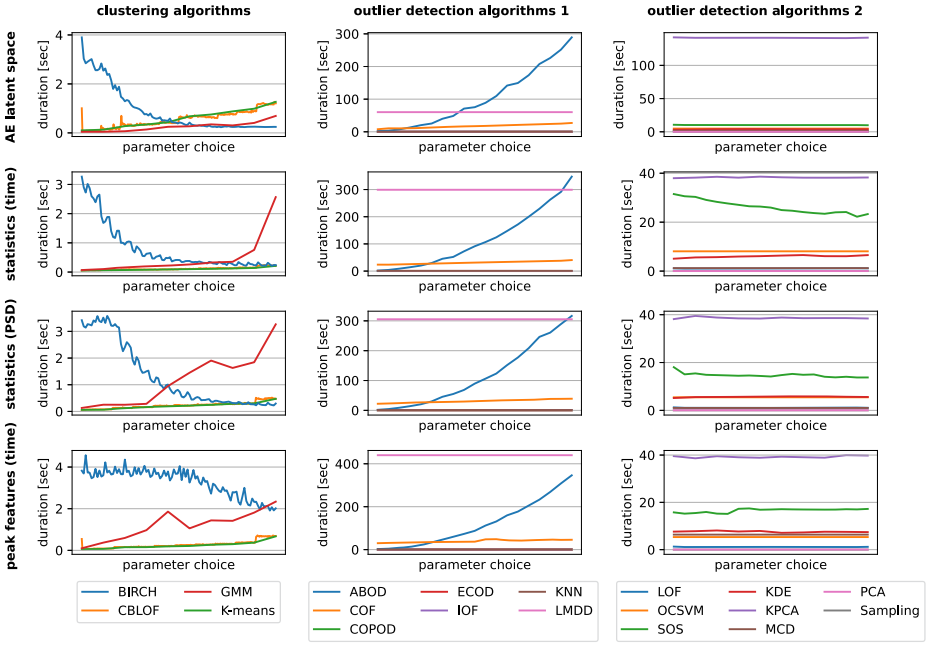

Figure 8. Fitting Durations

Figure 9 shows the duration needed by the algorithms to classify the validation data samples. The inference duration consists of both the feature extraction part and the algorithmic classification part.

The feature extraction methods are based on efficient signal processing algorithms, such as fast Fourier transforms or basic statistical calculations. Therefore, feature extraction has a small impact on the overall inference duration. The maximum allowed inference duration is 289.6 s. This criterion is fulfilled by all algorithms for all feature extraction methods and all parameter settings. All clustering algorithms perform particularly well, followed by the other algorithms and the outlier detection algorithms. For all feature extraction methods, ABOD has the worst inference duration when many nearest neighbors are used. The LMDD algorithm has the second highest inference duration.

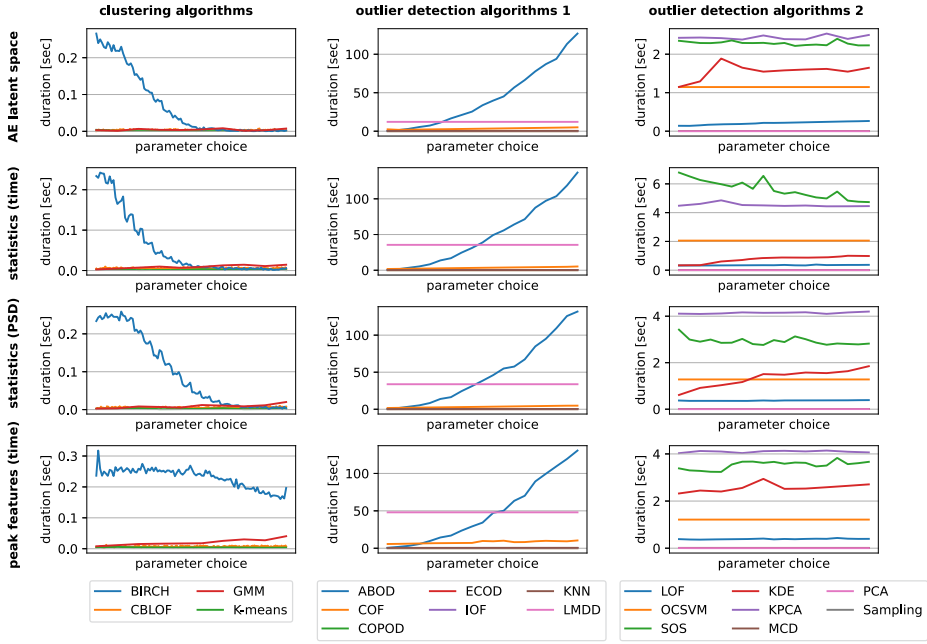

Figure 9. Fault Detection Algorithms Inference Durations

In the following, we describe the ability of the algorithms to classify disturbed data samples as disturbed and non disturbed data samples as normal. For both the fitting dataset and the validation dataset, we manually assigned a label to each data sample, either undisturbed or disturbed. The AUROC score is based on the manually assigned reference and the labels assigned by the fault detection algorithms.

5.4.1. Clustering Algorithms

The AUROC results and the condition specific accuracies of the clustering algorithms with respect to the feature extraction methods are depicted in Figure 10.

In general, features from the PSDs (statistics, and peak features) form a good basis for clustering algorithms to reliably identify disturbed laser oscillator feedback systems, since all clustering algorithms except the GMM achieve very good AUROC scores and high accuracies for all conditions. The GMM algorithm does not achieve satisfactory results for any combination of parameter setting and feature extraction method. It is noticeable that the condition specific accuracies obtained by GMM show that the GMM algorithm classifies all data samples as disturbed. From the results of the CBLOF algorithm, it can be seen that the fault detection quality is strongly dependent on the choice of input parameter. At an alpha of 0.5, the best AUROC scores of 1.0 are obtained on the fitting and validation dataset regardless of the choice of cluster number and beta for all feature extraction methods. Birch, and the K-Means algorithms achieve perfect results on the validation dataset for features from PSDs and the correct parameter choice.

The very good results of the clustering algorithms can be described with the help of the structure of the examined data. As the t-SNE embeddings of the data set already

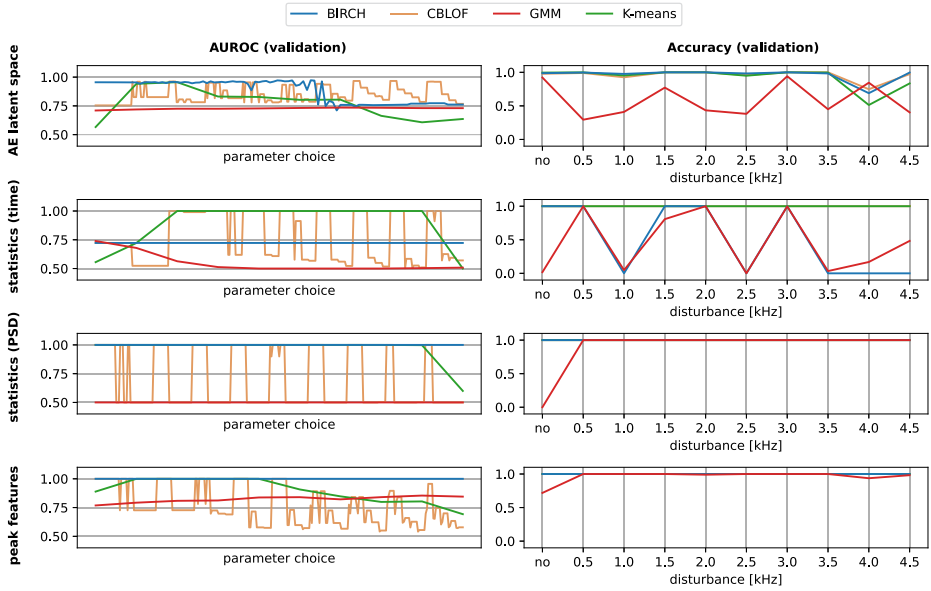

Figure 10. Results of clustering algorithms

indicate (see Figure 7), the data measured under similar conditions are positioned in cluster-like structures, especially using the PSD statistics and the peak features.

5.4.2. Outlier Detection Algorithms

The AUROC scores and the condition-dependent accuracies of the outlier detection algorithms are shown in Figure 11

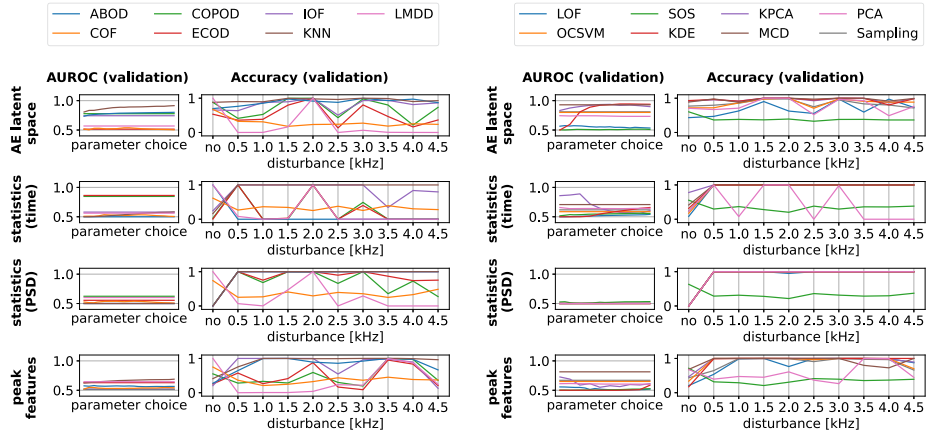

Figure 11. Results of the Outlier Detection Algorithms

In general, it can be seen that no combination of outlier detection algorithm and feature extraction method achieves a perfect AUROC score of 1.0. It is also noticeable

that the choice of parameters for the outlier detection algorithms has no great influence on the result, because the maximum AUROC scores hardly differ from the minimum AUROC scores per algorithm. In contrast to the clustering algorithms, the outlier detection algorithms achieve very poor AUROC scores on the feature datasets that use PSDs as a basis. Among all outlier detection algorithms, KNN achieves the highest AUROC score of 0.9148 using the AE latent space as features. The corresponding condition specific conditions show that the data recorded under no excitation are correctly classified with an accuracy of 0.8788. The accuracies that KNN detects an excited system from the controller data are all higher than 0.9.

It is noticeable that all algorithms that were fitted with PSD statistics are predictors that classify all validation data as disturbed. This implies that the algorithms cannot generalize the error detection learned on the PSD statistics fitting dataset because not all data samples from the fitting data set are classified as disturbed. Furthermore, it can be seen that similar to the outlier detection algorithms, none of the other algorithms achieve a perfect AUROC score of 1.0 on the validation dataset.

For all feature extraction methods KPCA achieves the highest AUROC scores, with the highest value of 0.94 being achieved with the AE latent space as the feature. The number of principal components leading to the highest AUROC scores for the respective feature extraction methods differ. Therefore, there exists no correct choice of principal components such that KPCA can describe the error detection behavior for all feature datasets. The second highest AUROC score on the validation dataset is also achieved on the AE latent space by the MCD algorithm.

In addition, the AE fault detector achieves AUROC scores of 1.0 on both the fitting and validation datasets making it a perfect fault detector.

5.4.3. Fault Classification Algorithms

Figure 12 presents the classification results of the classification algorithms, each with various parameter combinations using the four different feature extraction methods. The outcomes are showcased through fitting duration, inference duration on the validation dataset, and AUROC scores on a validation dataset. It can be seen that the AE latent space, among all feature extraction methods, serves as the best input for the selected classification algorithms for achieving good AUROC scores (> 0.95).

When utilizing the sigmoid function as a kernel, the SVM exhibits the lengthiest fitting and inference durations across all feature extraction methods. However, these durations remain within the acceptable threshold of 289.6 s. Among all algorithms and feature extraction methods, SVM achieves the highest AUROC score of 0.967 when employing the radial basis function (rbf) kernel and the AE latent space as the feature extraction method.

Of all the algorithms and feature extraction methods tested, the decision tree performs the poorest, while the random forest classifier emerges as the second-best performer. Additionally, the k-nearest neighbor classifier demonstrates improved AUROC scores as the number of analyzed neighbors increases.

5.4.4. Summary

Table 8 gives an overview of the fault detection algorithms and their parameter configuration that achieve an AUROC score higher than 0.95 on the validation dataset. If an

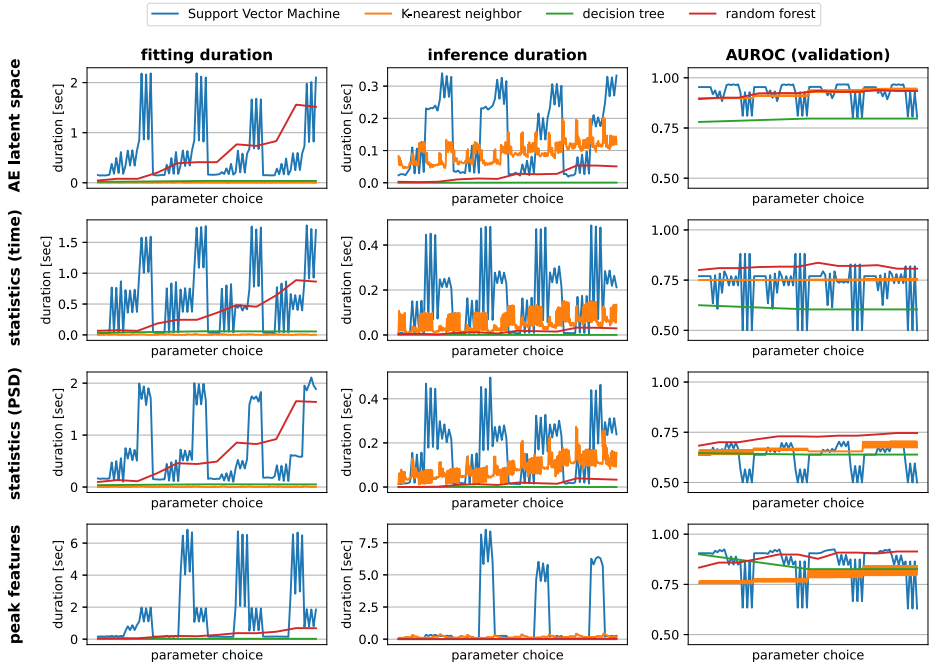

Figure 12. Results of the Fault Classification Algorithms

algorithm achieves such an AUROC score with multiple parameter combinations, we selected the parameter combination that gives the best AUROC score and the lowest inference duration.

The AE works directly on the PSDs. Therefore, no prior feature extraction is required. Among the algorithms that require prior feature extraction, only the clustering algorithms K-means clustering and CBLOF achieve very good AUROC scores on all validation datasets. Additionally, BIRCH achieves very good AUROC scores on the validation dataset using either the AE latent space, PSD statistics, or peak characteristics. Furthermore, it stands out that no algorithm which is fitted with the AE latent space achieves a perfect AUROC score. The best algorithms that do not belong to the clustering algorithms are KPCA having an AUROC score of 0.9368 and KDE with an AUROC score of 0.9436, both using the AE latent space as feature.

Table 9 shows for each feature extraction method the algorithm with the highest AUROC score on the validation dataset. The highest AUROC score is achieved by the SVM algorithm trained with the AE latent space. Statistics from PSDs do not provide a basis for reliable classification.

6. Conclusion

In this paper, we investigated the ability of data-based fault detection and fault classification algorithms in combination with four feature extraction methods to model the condition of an actively controlled phase-locked laser oscillator and determined the best methods and parameters for detecting and classifying disturbances that affect the healthy

Table 8. Best Fault Detection Results on the Validation Dataset

Feature	Algorithm	Parameter	AUROC	Inference duration in s
-	AE	-	**1.0**	0.1335
AE latent space	BIRCH	threshold: 2.4 branching factor: 80	0.9726	0.007
	K-means	# clusters: 4	0.9559	0.0014
	CBLOF	# clusters: 10 alpha: 0.5 beta: 5	0.9667	0.0018
statistics (time)	CBLOF	# clusters: 9 alpha: 0.6 beta: 1.5	**1.0**	0.002
	K-means	# clusters: 4	**1.0**	0.002
statistics (PSD)	BIRCH	threshold: 3.8 branching factor: 20	**1.0**	0.0019
	CBLOF	# clusters: 2 alpha: 0.5 beta: 1.5	**1.0**	**0.0018**
	K-means	# clusters: 5	**1.0**	0.0019
peak characteristics	BIRCH	threshold: 4.0 branching factor: 40	**1.0**	0.1606
	CBLOF	# clusters: 2 alpha: 0.7 beta: 1.5	**1.0**	**0.0018**
	K-means	# clusters: 6	**1.0**	0.0036

Table 9. Best Fault Classification Results on the Validation Dataset

Feature	Algorithm	Parameter	AUROC	Inference duration in s
AE latent space	svm	regularization parameter: 2 kernel: rbf	**0.9674**	0.2029
statistics (time)	svm	regularization parameter: 1 kernel: sigmoid	0.8811	0.2475
statistics (PSD)	RF	# estimators: 200 split criterion: entropy	0.7463	0.0362
peak characteristics	svm	regularization parameter: 1 kernel: poly	0.9246	0.0333

operation of the synchronization system. The fault detection methods were validated experimentally by disturbing the system acoustically. We evaluated the classification

performance for each combination of feature extraction, machine learning algorithm, and algorithmic-specific parameters using the fitting duration, inference duration, and AUROC scores as quality measures.

From the fault detection results, we can conclude that very good prediction results can be obtained without deep system expertise. Comparing the prediction results of the different types of algorithms, we notice that clustering algorithms achieve the best results regardless of the feature extraction methods. Moreover, there is no combination of an algorithm not belonging to the clustering algorithms and a feature extraction method that achieves a perfect AUROC score on the validation dataset. Additionally, there is no combination of a fault detection algorithm and the AE latent space as a feature extractor that achieves a perfect AUROC score on the validation dataset. With an AUROC score of 1.0 and a inference duration of 0.0018 s when applied to the validation dataset, the combination of CBLOF and peak characteristics or the combination of CBLOF and statistics from PSDS achieve the best results. However, we would like to draw particular attention to the performance of the AE fault detector, as it does not require prior feature extraction and can thus be applied directly to any dynamic system controlled in a closed loop. In addition, the inference time for the validation dataset is below the maximum acceptable threshold for real-time fault detection.

The results of fault classification highlight the impressive predictive capabilities of data-driven approaches. Notably, all algorithms have their best predictive performance when using the AE latent space feature extraction, a contrast to the fault prediction outcomes. The support vector machine trained on the AE latent space representation of the PSD emerges as the most effective classification algorithm among them.

In this paper, we compared the performance of the four feature extraction methods separately from each other. It would make sense to merge these feature extraction methods and train machine learning models on a combined feature dataset. The experimental evaluation used in this work is based on the excitation of different frequencies at the same level by a surface loudspeaker. For future work, we plan to investigate what minimum interference intensity must be present for a fault detection algorithm to be effective and to extend the fault detection mechanism to a predictive maintenance module that can predict when the next faulty operating point will occur.

References

[1] Sobolev E, Zolotarev S, Giewekemeyer K, Bielecki J, Okamoto K, Reddy HKN, et al. Megahertz Single-Particle Imaging at the European XFEL. Communications Physics. 2020;3(1).

[2] Schulz S, Czwalinna M, Felber M, Fenner M, Gerth C, Kozak T, et al., editors. Few-Femtosecond Facility-Wide Synchronization of the European XFEL: JACoW Publishing, Geneva, Switzerland; 2019.

[3] Heuer M. Identification and Control of the Laser-Based Synchronization System for the European X-ray Free Electron Laser [doctoralThesis]. Technische Universität Hamburg-Harburg; 2018.

[4] Welch P. The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms. IEEE Transactions on Audio and Electroacoustics. 1967;15(2):70-3.

[5] Zheng H, Wang R, Yin J, Li Y, Lu H, Xu M. A New Intelligent Fault Identification Method Based on Transfer Locality Preserving Projection for Actual Diagnosis Scenario of Rotating Machinery. Mechanical Systems and Signal Processing. 2020;135:106344.

[6] Siahpour S, Li X, Lee J. Deep Learning-Based Cross-Sensor Domain Adaptation for Fault Diagnosis of Electro-Mechanical Actuators. International Journal of Dynamics and Control. 2020;8(4):1054-62.

[7] Bagheri M, Zollanvari A, Nezhivenko S. Transformer Fault Condition Prognosis Using Vibration Signals Over Cloud Environment. IEEE Access. 2018;6:9862-74.
[8] Gamal M, Donkol A, Shaban A, Costantino F, Di G, Patriarca R. Anomalies Detection in Smart Manufacturing Using Machine Learning and Deep Learning Algorithms. In: Proceedings of the International Conference on Industrial Engineering and Operations Management, Rome, Italy; 2021. p. 1611-22.
[9] Lopez F, Saez M, Shao Y, Balta EC, Moyne J, Mao ZM, et al. Categorization of Anomalies in Smart Manufacturing Systems to Support the Selection of Detection Mechanisms. IEEE Robotics and Automation Letters. 2017;2(4):1885-92.
[10] Quevedo J, Puig V, Escobet T. Model Fault Detection of Feedback Systems: How and Why to Use the Output of the PID Controller? IFAC Proceedings Volumes. 2000;33(4):319-24. IFAC Workshop on Digital Control: Past, Present and Future of PID Control, Terrassa, Spain, 5-7 April 2000.
[11] Bauer M, Auret L, Bacci di Capaci R, Horch A, Thornhill NF. Industrial PID Control Loop Data Repository and Comparison of Fault Detection Methods. Industrial & Engineering Chemistry Research. 2019;58(26):11430-9.
[12] Wang Q, Liu J, Wei B, Chen W, Xu S. Investigating the Construction, Training, and Verification Methods of k-Means Clustering Fault Recognition Model for Rotating Machinery. IEEE Access. 2020;8:196515-28.
[13] Duong BP, Kim JM. Non-Mutually Exclusive Deep Neural Network Classifier for Combined Modes of Bearing Fault Diagnosis. Sensors. 2018;18(4).
[14] Kim D, Heo TY. Anomaly Detection with Feature Extraction Based on Machine Learning Using Hydraulic System IoT Sensor Data. Sensors. 2022;22(7).
[15] Li H, yun Xiao D. Fault Diagnosis Based on Power Spectral Density Basis Transform. Journal of Vibration and Control. 2015;21(12):2416-33.
[16] Wang Z, Yang J, Li H, Zhen D, Xu Y, Gu F. Fault Identification of Broken Rotor Bars in Induction Motors Using an Improved Cyclic Modulation Spectral Analysis. Energies. 2019;12(17).
[17] Sundaram S, Zeid A. Smart Prognostics and Health Management (SPHM) in Smart Manufacturing: An Interoperable Framework. Sensors. 2021;21(18).
[18] Christ M, Braun N, Neuffer J, Kempa-Liehr AW. Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package). Neurocomputing. 2018;307:72-7.
[19] Grünhagen A, Tropmann-Frick M, Eichler A, Fey G. Predictive Maintenance for the Optical Synchronization System of the European XFEL: A Systematic Literature Survey. In: BTW 2023. Bonn: Gesellschaft für Informatik e.V.; 2023. p. 1023-45.
[20] Tschannen M, Bachem O, Lucic M. Recent Advances in Autoencoder-Based Representation Learning. arXiv preprint arXiv:181205069. 2018.
[21] Xu R, Wunsch D. Survey of Clustering Algorithms. IEEE Transactions on Neural Networks. 2005;16(3):645-78.
[22] Zhang T, Ramakrishnan R, Livny M. BIRCH: An Efficient Data Clustering Method for Very Large Databases. SIGMOD Rec. 1996 jun;25(2):103-14.
[23] He Z, Xu X, Deng S. Discovering Cluster-Based Local Outliers. Pattern Recognition Letters. 2003;24(9):1641-50.
[24] Reynolds DA. Gaussian Mixture Models. Encyclopedia of biometrics. 2009;741(659-663).
[25] Hartigan JA, Wong MA. Algorithm AS 136: A K-Means Clustering Algorithm. Journal of the Royal Statistical Society Series C (Applied Statistics). 1979;28(1):100-8.
[26] Singh K, Upadhyaya S. Outlier Detection: Applications and Techniques. International Journal of Computer Science Issues (IJCSI). 2012;9(1):307.
[27] Kriegel HP, Schubert M, Zimek A. Angle-Based Outlier Detection in High-Dimensional Data. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '08. New York, NY, USA: Association for Computing Machinery; 2008. p. 444-52.
[28] Wang Y, Li K, Gan S. A Kernel Connectivity-based Outlier Factor Algorithm for Rare Data Detection in a Baking Process. IFAC-PapersOnLine. 2018;51(18):297-302. 10th IFAC Symposium on Advanced Control of Chemical Processes ADCHEM 2018.
[29] Li Z, Zhao Y, Botta N, Ionescu C, Hu X. COPOD: Copula-Based Outlier Detection. In: 2020 IEEE International Conference on Data Mining (ICDM); 2020. p. 1118-23.
[30] Li Z, Zhao Y, Hu X, Botta N, Ionescu C, Chen G. ECOD: Unsupervised Outlier Detection Using Empirical Cumulative Distribution Functions. IEEE Transactions on Knowledge and Data Engineering. 2022:1-1.

[31] Liu FT, Ting KM, Zhou ZH. Isolation-Based Anomaly Detection. ACM Trans Knowl Discov Data. 2012 mar;6(1).
[32] Parzen E. On Estimation of a Probability Density Function and Mode. The Annals of Mathematical Statistics. 1962;33(3):1065 1076. Available from: https://doi.org/10.1214/aoms/1177704472.
[33] Hoffmann H. Kernel PCA for Novelty Detection. Pattern Recognition. 2007;40(3):863-74.
[34] Angiulli F, Pizzuti C. Fast Outlier Detection in High Dimensional Spaces. In: European conference on principles of data mining and knowledge discovery. Springer; 2002. p. 15-27.
[35] Arning A, Agrawal R, Raghavan P. A Linear Method for Deviation Detection in Large Databases. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. KDD'96. AAAI Press; 1996. p. 164-9.
[36] Breunig MM, Kriegel HP, Ng RT, Sander J. LOF: Identifying Density-Based Local Outliers. SIGMOD Rec. 2000 may;29(2):93-104.
[37] Rousseeuw PJ, Driessen KV. A Fast Algorithm for the Minimum Covariance Determinant Estimator. Technometrics. 1999;41(3):212-23.
[38] Schölkopf B, Williamson R, Smola A, Shawe-Taylor J, Platt J. Support Vector Method for Novelty Detection. In: Proceedings of the 12th International Conference on Neural Information Processing Systems. NIPS'99. Cambridge, MA, USA: MIT Press; 1999. p. 582-8.
[39] Shyu ML, Chen SC, Sarinnapakorn K, Chang L. A Novel Anomaly Detection Scheme Based on Principal Component Classifier. Miami Univ Coral Gables Fl Dept of Electrical and Computer Engineering; 2003.
[40] Sugiyama M, Borgwardt K. Rapid Distance-Based Outlier Detection via Sampling. Advances in neural information processing systems. 2013;26.
[41] Janssens J, Huszár F, Postma E, van den Herik H. Stochastic Outlier Selection. Tilburg centre for Creative Computing, techreport. 2012;1:2012.
[42] Kingsford C, Salzberg SL. What are Decision Trees? Nature biotechnology. 2008;26(9):1011-3.
[43] Aha W, Kibler D, Albert M. Instance-Based Learning Algorithms. Machine Learning. 1991 01;6:37-66.
[44] Breiman L. Random forests. Machine learning. 2001;45:5-32.
[45] Platt J, et al. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. Advances in large margin classifiers. 1999;10(3):61-74.
[46] Zhao Y, Nasrullah Z, Li Z. PyOD: A Python Toolbox for Scalable Outlier Detection. Journal of Machine Learning Research. 2019;20(96):1-7.
[47] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011;12:2825-30.
[48] Goldin DQ, Kanellakis PC. On Similarity Queries for Time-Series Data: Constraint Specification and Implementation. In: Montanari U, Rossi F, editors. Principles and Practice of Constraint Programming — CP '95. Berlin, Heidelberg: Springer Berlin Heidelberg; 1995. p. 137-53.
[49] Van der Maaten L, Hinton G. Visualizing Data Using T-SNE. Journal of machine learning research. 2008;9(11).
[50] Fawcett T. An Introduction to ROC Analysis. Pattern Recognition Letters. 2006;27(8):861-74. ROC Analysis in Pattern Recognition.