

# Distributed and parallel time series feature extraction for industrial big data applications

**Maximilian Christ**

MAXIMILIAN.CHRIST@BLUE-YONDER.COM

*Blue Yonder GmbH, Karlsruhe, Germany*

**Andreas W. Kempa-Liehr**

KEMPA-LIEHR@FMF.UNI-FREIBURG.DE

*Freiburg Materials Research Center, University of Freiburg, Freiburg, Germany*

*Blue Yonder GmbH, Karlsruhe, Germany*

**Michael Feindt**

MICHAEL.FEINDT@BLUE-YONDER.COM

*on leave of absence from Karlsruhe Institute of Technology*

*Blue Yonder GmbH, Karlsruhe, Germany*

## Abstract

The all-relevant problem of feature selection is the identification of all strongly and weakly relevant attributes. This problem is especially hard to solve for time series classification and regression in industrial applications such as predictive maintenance or production line optimization, for which each label or regression target is associated with several time series and meta-information simultaneously. Here, we are proposing an efficient, scalable feature extraction algorithm, which filters the available features in an early stage of the machine learning pipeline with respect to their significance for the classification or regression task, while controlling the expected percentage of selected but irrelevant features.

The proposed algorithm combines established feature extraction methods with a feature importance filter. It has a low computational complexity, allows to start on a problem with only limited domain knowledge available, can be trivially parallelized, is highly scalable and based on well studied non-parametric hypothesis tests. We benchmark our proposed algorithm on all binary classification problems of the UCR time series classification archive as well as time series from a production line optimization project and simulated stochastic processes with underlying qualitative change of dynamics.

## 1. Introduction

Promising fields of application for machine learning are the Internet of Things (IoT) [Gubbi et al., 2013] and Industry 4.0 [Hermann et al., 2016] environments. In these fields, machine learning models anticipate future device states by combining knowledge about device attributes with historic sensor time series. They permit the classification of devices (e.g. hard drives) into risk classes with respect to a specific defect [Moblely, 2002]. Both fields are driven by the availability of cheap sensors and advancing connectivity between devices, which increases the need for machine learning on temporally annotated data.

In most cases the volume of the generated time series data forbids their transport to centralized databases [Gubbi et al., 2013]. Instead, algorithms for an efficient reduction of the data volume by means of feature extraction and feature selection are needed [Bolón-Canedo et al., 2015, p. 125–136]. Furthermore, for online applications of machine learning

it is important to continuously select relevant features in order to deal with concept drifts caused by qualitative changes of the underlying dynamics [Liu and Setiono, 1998].

Therefore, for industrial and other applications, one needs to combine distributed feature extraction methods with a scalable feature selection, especially for problems where several time series and meta-information have to be considered per label/target [Kusiak and Li, 2011]. For time series classification, it proved to be efficient to apply comprehensive feature extraction algorithms and then filter the respective features [Fulcher and Jones, 2014].

Motivated by industrial applications for machine learning models Christ et al. [2016] we are extending the approach of Fulcher and Jones and propose **FeatuRe Extraction based on Scalable Hypothesis tests (FRESH)**. The algorithm characterizes time series with comprehensive and well-established feature mappings and considers additional features describing meta-information. In a second step, each feature vector is individually and independently evaluated with respect to its significance for predicting the target under investigation. The result of these tests is a vector of p-values, quantifying the significance of each feature for predicting the label/target. This vector is evaluated on basis of the Benjamini-Yekutieli procedure [Benjamini and Yekutieli, 2001] in order to decide which features to keep.

The proposed algorithm is evaluated on all binary classification problems of the UCR time series classification archive [Chen et al., 2015] as well as time series data from a production line optimization project and simulated time series from a stochastic process with underlying qualitative change of dynamics [Liehr, 2013]. The results are benchmarked against well-established feature selection algorithms like linear discriminant analysis [Fulcher and Jones, 2014] and the Boruta algorithm [Kursa and Rudnicki, 2011], but also against Dynamic Time Warping [Wang et al., 2013]. The analysis shows that the proposed method outperforms Boruta based feature selection approaches as well as Dynamic Time Warping based approaches for problems with large feature sets and large time series samples. This contribution closes with a summary and an outlook on future work.

## 2. Time series feature extraction

Temporally annotated data come in three different variants [Elmasri and Lee, 1998]: Temporally invariant information (e.g. the manufacturer of a device), temporally variant information, which change irregularly (e.g. process states), and temporally variant information with regularly updated values (e.g. sensor measurements). The latter describe the continuously changing state  $s_{i,j}(t)$  of a system or device  $s_i$  with respect to a specific measurement of sensor  $j$ , which is repeated in intervals of length  $\Delta_t$ . This sampling captures the state of the system or device under investigation as a sequence

$$s_{i,j}(t_1) \rightarrow s_{i,j}(t_2) \rightarrow \dots \rightarrow s_{i,j}(t_\nu) \rightarrow \dots \rightarrow s_{i,j}(t_{n_t})$$

with  $t_{\nu+1} = t_\nu + \Delta_t$ . Such kind of sequences are called time series and are abbreviated by

$$\begin{aligned} \mathbf{s}_{i,j} &= (s_{i,j}(t_1), s_{i,j}(t_2), \dots, s_{i,j}(t_\nu), \dots, s_{i,j}(t_{n_t}))^\top \\ &= (s_{i,j,1}, s_{i,j,2}, \dots, s_{i,j,\nu}, \dots, s_{i,j,n_t})^\top. \end{aligned}$$

Here, we are considering  $i = 1, \dots, m$  devices with  $j = 1, \dots, n$  different time series per device. Therefore, we are dealing with  $n \cdot m \cdot n_t$  values describing the ensemble under

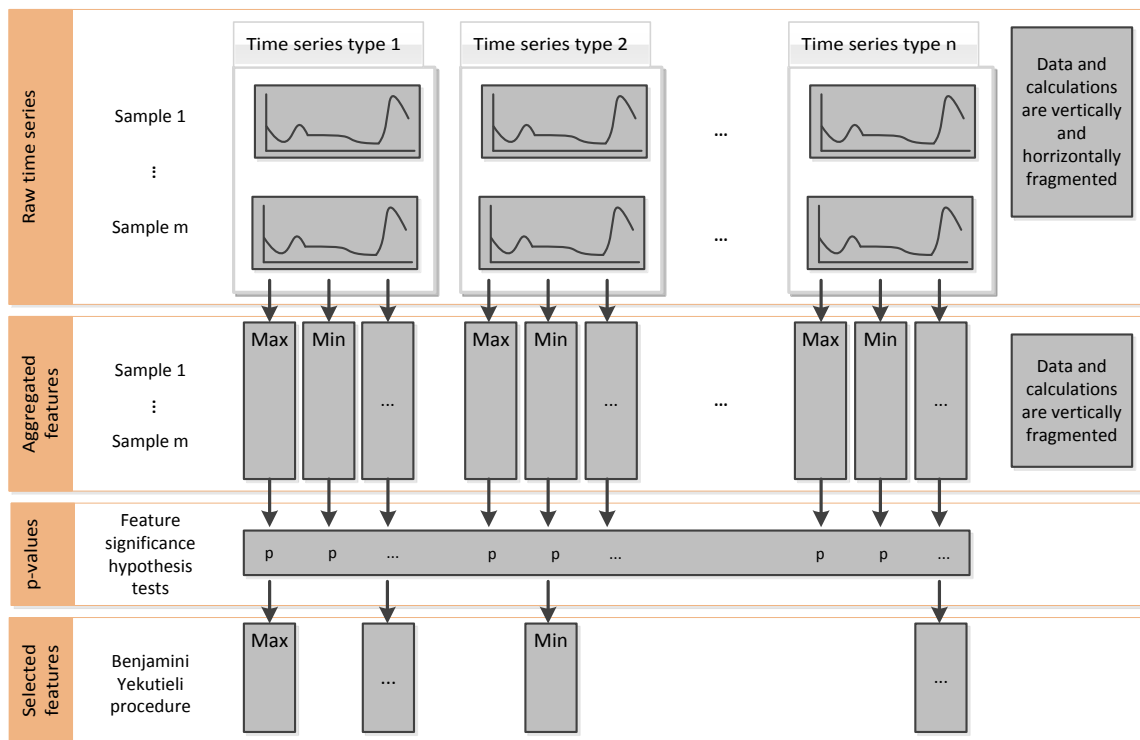


Figure 1: Data processing tiers of the filtered feature extraction algorithm. All but the Benjamini-Yekutieli procedure can be computed in parallel.

investigation. In order to characterize a time series with respect to its dynamics and reduce the data volume, a mapping  $\theta_k : \mathbb{R}^{n_t} \rightarrow \mathbb{R}$  is introduced, which captures a specific aspect  $k$  of the time series. One example for such mapping might be the maximum operator

$$\theta_{\max}(s_{i,j}) = \max\{s_{i,j,1}, s_{i,j,2}, \dots, s_{i,j,\nu}, \dots, s_{i,j,n_t}\},$$

which quantifies the maximal value ever recorded for time series  $s_{i,j}$ . This kind of lower dimensional representation is called a *feature*, which is a measurable characteristics of the considered time series. Other examples for feature mappings  $\theta_k$  of time series might be their mean, the number of peaks with a certain steepness, their periodicity, a global trend, etc. Comprehensive collections of time series feature mappings are discussed by [Fulcher and Jones \[2014\]](#) and [Nun et al. \[2015\]](#). [Fulcher and Jones \[2014\]](#) propose to use more than 9000 features from 1000 different feature generating algorithms.

Now, consider  $n_f$  different time series feature mappings, which are applied to all  $m \cdot n$  time series recorded from  $n$  sensors of  $m$  devices (Fig. 1). The resulting feature matrix  $\mathbb{X} \in \mathbb{R}^{m \times n_\phi}$  has  $m$  rows (one for each device) and  $n_\phi = n \cdot n_f + n_i$  columns with  $n_i$  denoting the number of features generated from device specific meta-information. Each column of  $\mathbb{X}$  comprises a vector  $X \in \mathbb{R}^m$  capturing a specific characteristic of all considered devices.

### 3. Feature filtering

Typically, time series are noisy and contain redundancies. Therefore, one should keep the balance between extracting meaningful but probably fragile features and robust but probably non-significant features. Some features such as the *median* will not be heavily influenced by outliers, others such as *max* will be intrinsically fragile. The choice of the right time series feature mappings is crucial to capture the right characteristics for the task at hand.

#### 3.1. Relevance of features

A meaningless feature describes a characteristic of the time series that is not useful for the classification or regression task at hand. Radivojac et al. [2004] considered a binary target  $Y$ , stating that the relevance of feature  $X$  is measured as the difference between the class conditional distributions  $f_{X|Y=0}$  and  $f_{X|Y=1}$ . We adopt this definition and consider a feature  $X$  being relevant for the classification of the binary target  $Y$  if those distributions are not equal. In general, a feature  $X$  is relevant for predicting target  $Y$  if and only if

$$\exists y_1, y_2 \text{ with } f_Y(y_1) > 0, f_Y(y_2) > 0 : f_{X|Y=y_1} \neq f_{X|Y=y_2}. \quad (1)$$

The condition from Equation (1) is equivalent to

$$\begin{aligned} & X \text{ is not relevant for target } Y \\ \Leftrightarrow & \forall y_1, y_2 \text{ with } f_Y(y_1) > 0, f_Y(y_2) > 0 : f_{X|Y=y_1} = f_{X|Y=y_2} \\ \Leftrightarrow & \forall y_1 \text{ with } f_Y(y_1) > 0 : f_{X|Y=y_1} = f_X \quad \Leftrightarrow \quad f_{X,Y} = f_{X|Y}f_Y = f_Xf_Y \\ \Leftrightarrow & X, Y \text{ are statistically independent} \end{aligned} \quad (2)$$

We will use the statistical independence to derive a shorter definition of a relevant feature:

**Definition 1 (A relevant feature)** *A feature  $X_\phi$  is relevant or meaningful for the prediction of  $Y$  if and only if  $X_\phi$  and  $Y$  are not statistically independent.*

#### 3.2. Hypothesis tests

For every extracted feature  $X_1, \dots, X_\phi, \dots, X_{n_\phi}$  we will deploy a singular statistical test checking the hypotheses

$$H_0^\phi = \{X_\phi \text{ is irrelevant for predicting } Y\}, H_1^\phi = \{X_\phi \text{ is relevant for predicting } Y\}. \quad (3)$$

The result of each hypothesis test  $H_0^\phi$  is a so-called p-value  $p_\phi$ , which quantifies the probability that feature  $X_\phi$  is not relevant for predicting  $Y$ . Small p-values indicate features, which are relevant for predicting the target.

Based on the vector  $(p_1, \dots, p_{n_\phi})^T$  of all hypothesis tests, a multiple testing approach will select the relevant features (Sec. 3.3). We propose to treat every feature uniquely by a different statistical test, depending on whether the codomains of target and feature are binary or not. The usage of one general feature test for all constellations is not recommended. Specialized hypothesis tests yield a higher statistical power due to more assumptions about the codomains that can be used during the construction of those tests. The proposed feature significance tests are based on nonparametric hypothesis tests, that do not make

any assumptions about the distribution of the variables, which ensures robustness of the procedure.

**Exact Fisher test of independence:** This feature significance test can be used if both the target and the inspected feature are binary. Fisher’s exact test [Fisher, 1922] is based on the contingency table formed by  $X_\phi$  and  $Y$ . It inspects if both variables are statistically independent, which corresponds to the hypotheses from Eq. (3).

Fisher’s test belongs to the class of exact tests. For such tests, the significance of the deviation from a null hypothesis (e.g., the p-value) can be calculated exactly, rather than relying on asymptotic results.

**Kolmogorov-Smirnov test (binary feature):** This feature significance test assumes the feature to be binary and the target to be continuous. In general, the Kolmogorov-Smirnov (KS) test is a non-parametric and stable goodness-of-fit test that checks if two random variables  $A$  and  $B$  follow the same distribution [Massey, 1951]:

$$H_0 = \{f_A = f_B\}, H_1 = \{f_A \neq f_B\}.$$

By conditionally modeling the distribution function of target  $Y$  on the two possible values  $x_1, x_2$  of the feature  $X_\phi$  we can use the KS test to check if the distribution of  $Y$  differs given different values of  $X_\phi$ . Setting  $A = Y|X_\phi = x_1$  and  $B = Y|X_\phi = x_2$  results in

$$H_0^\phi = \{f_{Y|X_\phi=x_1} = f_{Y|X_\phi=x_2}\}, H_1^\phi = \{f_{Y|X_\phi=x_1} \neq f_{Y|X_\phi=x_2}\}. \quad (4)$$

The hypotheses from Eq. (3) and (4) are equivalent as demonstrated in the chain of Equations in (2). Hence, the KS test can address the feature relevance of  $X_\phi$ .

**Kolmogorov-Smirnov test (binary target):** When the target is binary and the feature non-binary, we can deploy the Kolmogorov-Smirnov test again. We have to switch roles of target and feature variable, resulting in the testing of the following hypothesis:

$$H_0^\phi = \{f_{X_\phi|Y=y_1} = f_{X_\phi|Y=y_2}\}, H_1^\phi = \{f_{X_\phi|Y=y_1} \neq f_{X_\phi|Y=y_2}\}.$$

This time  $y_1$  and  $y_2$  are the two possible values of  $Y$  and  $f_{X_\phi|Y=y_j}$  is the conditional density function of  $X_\phi$  given  $Y$ . This hypothesis is also equivalent to the one in Eq. (3).

**Kendal rank test:** This filter can be deployed if neither target nor feature are binary. Kendall’s rank test [Kendall, 1938] checks if two continuous variables may be regarded as statistically dependent, hence naturally fitting our hypotheses from Eq. (3). It is a non-parametric test based on Kendall’s rank statistic  $\tau$ , measuring the strength of monotonic association between  $X_\phi$  and  $Y$ . The calculation of the rank statistic is more complex when ties are involved [Adler, 1957], i.e. feature or target are categorical.

### 3.3. Feature significance testing

When comparing multiple hypotheses simultaneously, errors in the inference tend to accumulate [Curran-Everett, 2000]. In this context, a wrongly added feature is a feature  $X_\phi$  for which the null hypothesis  $H_0^\phi$  has been rejected by the respective feature significance test, even though  $H_0^\phi$  is true. If we want to control the percentage of irrelevant added features we have to control the percentage of wrongly rejected null hypothesis among all hypothesis.

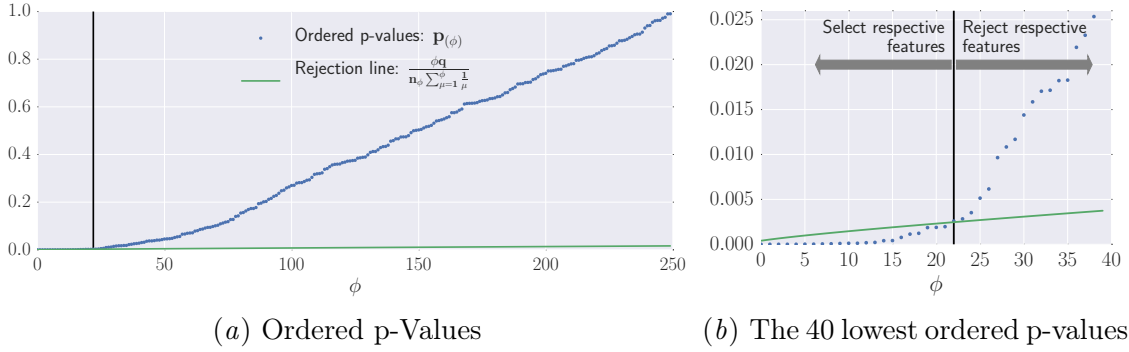


Figure 2: The Benjamini-Yekutieli procedure for a sample of simulated p-Values of 250 individual feature significance tests. The rejection line aims to control a FDR level  $q$  of 10%.

In multiple testing, this ratio of the expected proportion of erroneous rejections among all rejections is called false discovery rate (FDR).

The FDR as a measure of the accumulated statistical error was suggested by [Benjamini and Hochberg \[1995\]](#). Later the non-parametric Benjamini-Yekutieli procedure was proposed. Based on the p-values it tells which hypotheses to reject while still controlling the FDR under any dependency structure between those hypotheses [[Benjamini and Yekutieli, 2001](#)]. It will be the last component of our filtered feature extraction algorithm.

The procedure searches for the first intersection between the ordered sequence of p-values  $p_{(\phi)}$  (dotted blue curves in Fig. 2) with a linear sequence (green lines in Fig. 2)

$$r_{\phi} = \frac{\phi q}{n_{\phi} \sum_{\mu=1}^{\phi} \frac{1}{\mu}}. \quad (5)$$

Here,  $n_{\phi}$  is the number of all null hypotheses and  $q$  is the FDR level that the procedure controls. It will reject all hypotheses belonging to p-values that have a lower value than the p-value at the intersection, see the left side of Fig. 2(b).

### 3.4. The proposed feature extraction algorithm

We propose **FeatuRe Extraction based on Scalable Hypothesis tests (FRESH)** for parameter  $q \in [0, 1]$ , given by the following three steps:

1. Perform a set of  $n_{\phi}$  univariate feature mappings as introduced in Sec. 2 on  $m \cdot n$  different time series to create the features  $X_{\phi}$ ,  $\phi = 1, \dots, n_{\phi}$ .
2. For each generated feature  $X_1, \dots, X_{n_{\phi}}$  perform exactly one hypothesis test for the hypothesis  $H_0^{\phi}$  from Equation (3). To do so, take the corresponding feature significance test from Sec. 3.2. Calculate the p-values  $p_1, \dots, p_{n_{\phi}}$  of the tests.

3. Perform the Benjamini-Yekutieli procedure under correction for dependent hypotheses [Benjamini and Yekutieli, 2001] for a FDR level of  $q$  on the collected p-values  $p_1, \dots, p_{n_\phi}$  to decide which null hypothesis  $H_0^\phi$  to reject (c.f. Sec. 3.3). Only return features  $X_\phi$  for which the respective hypothesis  $H_0^\phi$  was rejected by the procedure.

### 3.5. Variants of FRESH

A problem of many filter feature methods such as the one we utilize in steps 2 and 3 of FRESH is the redundancy in the feature selection. As long as features are considered associated with the target, they will all be selected by the filter even though many of them are highly correlated to each other [Kira and Rendell, 1992]. For example *median* and *mean* are highly correlated in the absence of outliers and therefore we expect FRESH to either select or drop both *median* and *mean* at the same time. To avoid generating a group of highly correlated features we propose to add another step to FRESH:

- \*. Normalize the features and perform a principal performance analysis (PCA). Keep the principal components with highest eigenvalue describing  $p$  percent of the variance.

This step will reduce the number of features and the obtained principal components are de-correlated, orthogonal variables [Hotelling, 1933].

One could perform step \* between steps 1 and 2 of FRESH to get rid of the correlations between the created variables early. Then the feature significance tests in step 2 of FRESH will take principal components instead of the original features as input. We will denote this variant of FRESH as FRESH\_PCAb(efore). Also, one could perform step \* after the FRESH algorithm, directly after step 3. This means that the PCA will only process those features, which are found relevant by the FRESH algorithm instead of processing all features. This variant of FRESH is called FRESH\_PCAa(fter).

## 4. Evaluation

In the following, the performance of FRESH, its two variants from Sec. 3.5 and other time series feature extraction methods are compared. The evaluation is done with respect to both the meaningfulness of the extracted features as well as the time it takes to extract the features. While doing so, all feature extraction methods operate on the same feature mappings, the differences lay only in the used feature selection process.

### 4.1. Setup

FRESH is parameterized with  $q = 10\%$ , cf. Eq. (5). Its variants, which apply a PCA, are using  $p = 95\%$  (Sec. 3.5). Full\_X uses all the features, which are created during step 1 of FRESH (Sec. 3.4) without any subsequent filtering. Features contained in Full\_X will be filtered by applying the Boruta feature selection algorithm [Kursa and Rudnicki, 2011], or by a forward selection with a linear discriminant analysis classifier denoted LDA. Further, the direct classifier DTW\_NN, a nearest neighbor search under the Dynamic Time Warping distance, is considered [Wang et al., 2006]. Those six different extraction methods and DTW\_NN were each picked for a reason. DTW\_NN is reported to reach the highest accuracy rates among other time series classifiers, LDA was the first proposed algorithm to automatically



extract features from time series [Fulcher and Jones, 2014] and `Boruta` is a promising feature selection algorithm that incorporates interactions between features.

To guarantee reproducibility we use all 31 time series data sets from the UCR time series archive [Chen et al., 2015] containing a binary classification problem. To compare the runtime of the different methods, time series of flexible length and sample number belonging to two classes are generated by simulating the stochastic dynamics of a dissipative soliton [Liehr, 2013, p. 164]. The last data source originates from the production of steel billets, extracted during the German research project `iPRODIGT`. The project demonstrates a typical application of industrial time series analysis, aiming to predict the passing or failing of product specification testings based on timely annotated data. It contains 26 univariate meta-variables forming the baseline feature set extended by 20 different sensor time series having up to 44 data points for each sample. The data set contained 1554 samples of two classes "*broken*" and "*not broken*" each, in total 3108 samples.

Due to limitations regarding the size of this paper, we do not address the regression performance of `FRESH` and variants yet. In a future work, we catch up on this.

## 4.2. Accuracy

For the data sets from the UCR time series repository as well as the `iPRODIGT` data, the underlying structure and therefore the relevant features are unknown. We cannot compare the different methods on their ability to extract meaningful features because we do not know which features are meaningful and which are not. Also, we cannot compare the extracted features to direct classifiers such as `DTW_NN`. Therefore, we evaluate the performance of the feature extraction algorithms by comparing the performance of a classification algorithm on the extracted features. Hereby, we assume that more meaningful features will result in a better classification result.

To investigate the feature extraction methods under different conditions and to make sure that the feature filtering is not tuned to a specific classifier, the classification task is solved by a collection of a one layer neural network/perceptron (`NN`), a logistic regression model (`LR`), a support vector machine (`SVM`), a random forest classifier (`RFC`) and an adaboost classifier (`ABC`). The hyperparameters for those methods are not optimized to get an unbiased view on the meaningfulness of the extracted features, instead the default values from the Python package `scikitlearn` version 0.17.1 were used [Pedregosa et al., 2011].

For every data set, all available samples will be used to perform the feature extraction itself. Then, one third of the samples are randomly picked for a test set, the remaining two thirds are used to train the classifiers. We define the index set for the classification algorithms as  $\mathcal{M} := \{\text{NN, SVM, RFC, ABC, LR}\}$  and for the inspected feature extraction methods we define  $\mathcal{A} := \{\text{FRESH, LDA, FULL\_X, FRESH\_PCAa, FRESH\_PCAb, Boruta}\}$ . Then, the accuracy of a classifier  $m \in \mathcal{M}$  on the test part of the data set  $d$  for the features generated by method  $a \in \mathcal{A}$  is denoted as  $acc_m^d(a)$ .

Further, we calculate the mean of the accuracy of the five classification algorithms, which is denoted by  $acc^d(a) := \frac{1}{5} \sum_{m \in \mathcal{M}} acc_m^d(a)$ . `DTW_NN` itself does not perform any feature extraction and its accuracy is directly calculated by predicting on the test set. It is denoted by  $acc^d(\text{DTW\_NN})$ . Now, we are able to denote the average accuracy over all 31 UCR data



sets with binary classification tasks [Chen et al., 2015] for each method  $a \in \mathcal{A} \cup \{\text{DTW\_NN}\}$  by  $acc^{UCR}(a) = \frac{1}{31} \sum_{i=1, \dots, 31} acc^{d_i}(a)$ , cf. columns 3 and 7 of Tab. 1.

From the 3<sup>rd</sup> column of Tab. 1 we can observe that FRESH\_PCAa dominated the feature based approaches on the UCR data sets but it was not able to beat DTW\_NN. On the iPRO-DICT data, DTW\_NN could only operate on one type of time series without the univariate features. This seems to be the reasons why Boruta and FRESH\_PCAa beat the accuracy of DTW\_NN as shown in the 7<sup>th</sup> column of Tab. 1. Here FRESH\_PCAa again achieved the highest accuracy among all feature based approaches.

Further, we count how often a feature extraction method  $a^* \in \mathcal{A}$  reaches the highest accuracy for the 155 classifier/ data set combinations on the 31 UCR data sets among all six feature extraction methods in  $\mathcal{A}$  while a draw counts for both methods:

$$n_{best}^{UCR}(a^*) := \left| \left\{ (m, d_i) \mid m \in \mathcal{M}, i = 1, \dots, 31 : acc_m^{d_i}(a^*) = \max\{acc_m^{d_i}(a) \mid a \in \mathcal{A}\} \right\} \right|.$$

The evaluation metric  $n_{best}^{UCR}$  is reported in the 5<sup>th</sup> column of Tab. 1. Again FRESH\_PCAa achieved the best result, for over half of the 155 inspected combinations it had the highest reported accuracy and again Boruta came in second with 74.3 combinations on average. Regarding both accuracy metrics, FRESH\_PCAa seems to be favorable over the other considered feature based approaches, with Boruta coming close.

Table 1: Performance metrics for the different feature extraction methods and DTW\_NN on the 31 two-class data sets from the UCR time series archive as well as the data from the iPRO-DICT research project. All simulations have been conducted three times and the values of the performance metrics have been averaged to reduce the dependency on random elements such as the test and train data split.

Method	$\overline{t^{UCR}}$	$\overline{acc^{UCR}}$	$\overline{n_f^{UCR}}$	$\overline{n_{best}^{UCR}}$	$\overline{t^{project}} \cdot 10^3$	$\overline{acc^{project}}$	$\overline{n_f^{project}}$
DTW_NN	152.1 ± 0.3	0.926 ± 0.011	–	–	0.04 ± 0.016	0.552 ± 0.013	–
Full_X	33.4 ± 0.5	0.752 ± 0.018	161.00 ± 0.00	39.0 ± 3.6	3.30 ± 0.149	0.520 ± 0.003	3246 ± 0
FRESH	33.9 ± 0.5	0.838 ± 0.020	40.35 ± 0.05	61.0 ± 5.0	3.31 ± 0.148	0.546 ± 0.007	309 ± 52
FRESH_PCAa	33.9 ± 0.5	0.877 ± 0.010	8.22 ± 0.00	79.0 ± 3.0	3.31 ± 0.146	0.566 ± 0.017	33 ± 4
FRESH_PCAb	33.5 ± 0.5	0.760 ± 0.010	3.93 ± 0.00	19.0 ± 4.4	4.01 ± 0.276	0.515 ± 0.023	2 ± 1
LDA	37.5 ± 0.6	0.678 ± 0.020	27.23 ± 0.02	13.0 ± 1.7	474 ± 390	0.525 ± 0.010	229 ± 114
Boruta	104.5 ± 0.3	0.841 ± 0.014	63.29 ± 0.28	74.3 ± 6.7	4.64 ± 0.241	0.562 ± 0.006	104 ± 3

### 4.3. Runtime

The second and sixth column of Tab. 1 contains the average *pipeline runtime*, which is the combined runtime of feature extraction, training of a classifier and predicting. For DTW\_NN the pipeline runtime captures just the fitting and predicting steps as no features are extracted. There are tradeoffs between the number of extracted features and the time spent on feature extraction which force us to consider the pipeline runtime instead of the extraction runtime. E.g., time spent in the extraction process can be compensated by a lower number of extracted features which reduces the time spent for training the final classifier.

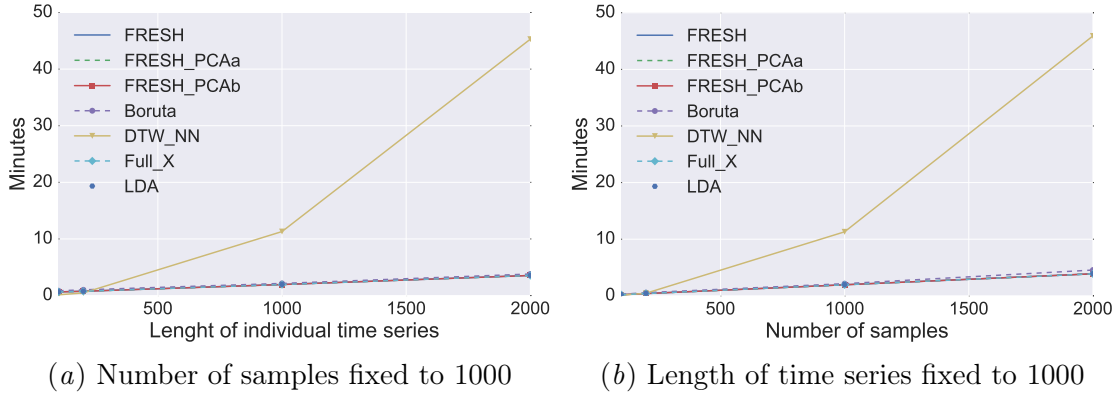


Figure 3: Average pipeline runtime of time series classification concerning the nonlinear dynamics of a dissipative soliton [Liehr, 2013, p. 164]. The curves of all methods except DTW\_NN lay on top of each other.

Analog to the accuracy evaluation metrics,  $t_m^d(a)$  denotes the pipeline runtime in seconds of classifier  $m \in \mathcal{M}$  and feature extraction method  $a \in \mathcal{A}$  on data set  $d$ . In the same way,  $t^d(a) = \frac{1}{5} \sum_{m \in \mathcal{M}} t_m^d(a)$  denotes the average pipeline runtime in seconds over all 5 classifiers and  $t^d(\text{DTW\_NN})$  the runtime of fitting and predicting by DTW\_NN. The average pipeline runtimes over all UCR data sets for each method  $a \in \mathcal{A} \cup \{\text{DTW\_NN}\}$  are denoted by  $t^{UCR}(a) = \frac{1}{31} \sum_{i=1, \dots, 31} t^{d_i}(a)$ . All calculations are executed on a single computational core in order to increase comparability.

The full feature matrix Full\_X is the fastest extraction algorithm in our comparisons on both the UCR and the iPRODIGT data, as seen in the 2<sup>nd</sup> and 6<sup>th</sup> column of Tab. 1. Saved time for the fitting of the feature selection algorithm compensated for the five classifiers having to be trained on more features. Accordingly, the PCA step of FRESH\_PCAa saves so much time for the fitting of the classifiers that this step is basically “free”, while FRESH\_PCAa has a mean pipeline runtime of 33.86 seconds. On average, FRESH takes 33.87 seconds. The same can be observed on the iPRODIGT data where Full\_X, FRESH and FRESH\_PCAa all had similar pipeline runtimes even though the number of extracted features varied greatly. The low average pipeline runtime of just  $t^{project} = 40.17$  seconds for DTW\_NN in the 6<sup>th</sup> column of Tab. 1 is due to the classification algorithm only considering one type of time series while the extraction methods operate on 20 different time series.

Apart from the observed runtimes, we are interested in the feature extraction method’s ability to scale with an increasing number of feature mappings, time series length and device numbers. As expected, Fig. 3 shows that all considered feature extraction methods – in contrast to DTW\_NN – scale linearly with an increasing length of the time series or increasing number of samples. This is due to the considered feature mapping having a linear runtime with respect to the length of the time series. However, Fig. 4 shows that, among the feature based approaches, only FRESH and FRESH\_PCAa scale linear with an increasing number of features (e.g. due to more devices, feature mappings or types of time series).

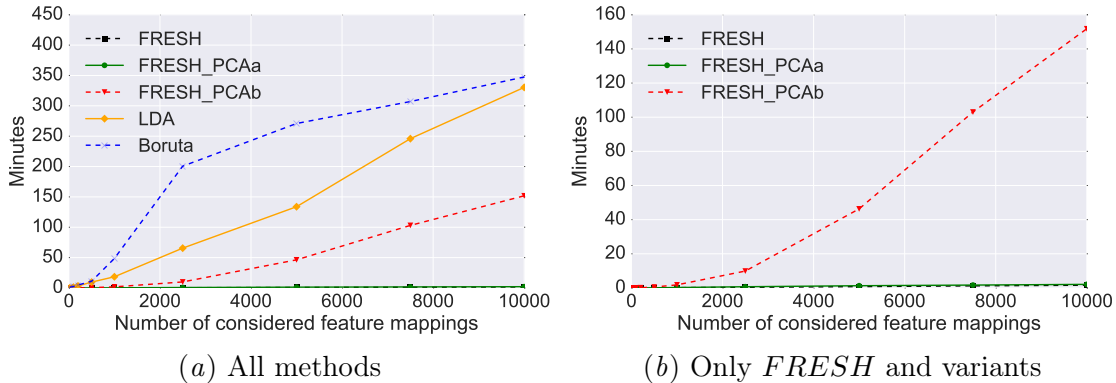


Figure 4: Average feature extraction runtime during ten feature selection runs for 10,000 samples of different amounts of simulated feature mappings (the curves of **FRESH** and **FRESH\_PCAa** are overlapping)

#### 4.4. Selected features

The number of features extracted by algorithm  $a \in \mathcal{A}$  on the data set  $d$  are denoted by  $n_f^d(a)$ . Again this can be averaged over the UCR data sets resulting in  $n_f^{UCR}(a) = \frac{1}{31} \sum_{i=1, \dots, 31} n_f^{d_i}(a)$ , cf. rightmost column of Tab. 1.

During our simulations, **FRESH\_PCAa** was able to reduce the number of features drastically. For the UCR data, it reduces 161 considered feature mappings to an average number of 8.2 features while on the iPRODIGT data it reduces the 3246 calculated features to 33. **FRESH\_PCAb** only selected four respective two features on average, which may explain its low accuracies. If one wishes to extract a minimal set of relevant features, we recommend to deploy **FRESH\_PCAa**, as it extracted the second lowest number of features but achieved the highest and second highest accuracies.

#### 4.5. Resume

We proposed **FRESH** as a highly scalable feature extraction algorithm. Our simulations showed that, in contrast to other considered methods, **FRESH** is able to scale with the number of feature mappings and samples as well as with the amount of different types and length of the time series. While doing so, it is extracting meaningful features as demonstrated by competitive accuracies.

The relative bad performance of **FRESH\_PCAb** seems to originate in the PCA step selecting features only based on their ability to explain the variance in the input variables and not in their significance to predict the target variable. By this, relevant information for the classification or regression task can get lost.

On the other hand, the combination of **FRESH** with a subsequent PCA filtering to reduce the number of redundant and highly correlated features, denoted as **FRESH\_PCAa** was overall the most competitive feature based method in our evaluation. On the UCR data, it achieved

the second highest and on the iPRODIGT data it reached the highest accuracy. Further, it had the second lowest number of extracted features.

## 5. Discussion

### 5.1. FRESH assists the acquisition of domain knowledge

It is common knowledge that the quality of feature engineering is a crucial success factor for supervised machine learning in general [Domingos, 2012, p. 82] and for time series analysis in particular [Timmer et al., 1993]. But comprehensive domain knowledge is needed in order to perform high quality feature engineering. Contrarily, it is quite common for machine learning projects that data scientists start with limited domain knowledge and improve their process understanding while continuously discussing their models with domain experts. This is basically the reason, why dedicated time series models are very hard to build from scratch.

Our experience with data science projects in the context of IoT and Industry 4.0 applications Christ et al. [2016] showed that it is very important to identify relevant time series features in an early stage of the project in order to engineer more specialized features in discussions with domain experts. The FRESH algorithm supports this approach by applying a huge variety of established time series feature mappings to different types of time series and meta-information simultaneously and identifies relevant features in a robust manner.

We observe that features extracted by FRESH contribute to a deeper understanding of the investigated problem, because each feature is intrinsically related to a distinct property of the investigated system and its dynamics. This fosters the interpretation of the extracted features by domain experts and allows for the engineering of more complex, domain specific features Christ et al. [2016] including dedicated time series models, such that their predictions in return might become a future feature mapping for FRESH.

### 5.2. FRESH is operational

We have already mentioned that FRESH has been developed in the course of IoT and Industry 4.0 projects Christ et al. [2016]. Especially for predictive maintenance applications with limited numbers of samples and high level of noise in e.g. sensor readings, it has been proven as crucial to filter irrelevant features in order to prevent overfitting. To ensure a robust and scalable filtering, we consider each feature importance individually. This causes several implications:

- FRESH is robust in the sense of classical statistics, because the hypothesis tests and the Benjamini-Yekutieli procedure do not make any assumptions about the probability distribution or dependence structure between the features. Here, robustness refers to the insensitivity of the estimator to outliers or violations in underlying assumptions [John et al., 2013].
- FRESH is not considering the meaningfulness of interactions between features by design. Hence, in its discussed form it will not find meaningful feature combinations such as chessboard variables [Guyon and Elisseeff, 2003, Fig. 3a]. However, in our evaluation process the feature selection algorithm Boruta, which considers feature interactions, was not able to beat the performance of FRESH. Further, it is possible for FRESH

to incorporate combinations of features and pre-defined interactions as new features themselves.

- FRESH is scalable due to the parallelity of the feature calculation and hypothesis tests (see the two topmost tiers in Fig. 1) and can be trivially parallelized and even distributed over several computational units. In addition, the feature filter process has computational costs compared to feature calculation and significance testing. Therefore, FRESH scales linearly with the number of extracted features, length of the time series, and number of considered time series.
- A side effect of ensuring robustness and testing features individually is that FRESH tends to extract highly correlated features, which could result in poor classification performance. We propose to combine FRESH with a subsequent PCA, which has been discussed as FRESH\_PCAa in Sec. 3.5 and indeed improved the performance significantly.

### 5.3. Feature selection of FRESH

Nilsson et al. [2007] proposed to divide feature selection into two flavors, the *minimal optimal problem* is finding a set consisting of all strongly relevant attributes and a subset of weakly relevant attributes such that all remaining weakly relevant attributes contain only redundant information. The *all-relevant problem* is finding all strongly and weakly relevant attributes. The first problem is way harder than the second, even asymptotically intractable for strictly positive distributions [Nilsson et al., 2007]. Accordingly, FRESH solves the second, easier problem as we extract every relevant feature, even though it might be a duplicate or highly correlated to another relevant feature [cf. Kursa and Rudnicki, 2011].

Yu and Liu [2003] separated feature selection algorithms into two categories, the *wrapper model* and the *filter model*. While the selection of wrapper models is based on the performance of a learning algorithm on the selected set of features, filter models use general characteristics to derive a decision about which features to keep. Filter models are further divided into feature weighting algorithms and subset search algorithms, which evaluate the goodness of features individually or through subsets. According to this definition, the feature selection part of FRESH is a filter model, more precisely, a feature weighting algorithm.

FRESH contains a feature selection part on basis of hypothesis tests and the Benjamini-Yekutieli procedure, which of course can be used as a feature selection algorithm itself. But, due to its systematic incorporation of scalable time series feature mappings and the proposed decomposition in computing tiers (Fig. 1) it is especially applicable to the needs of mass time series feature extraction and is considered as a feature extraction algorithm.

By applying a multiple testing algorithm, FRESH avoids the “*look-elsewhere effect*” [Gross and Vitells, 2010] which is a statistically significant observation arising by chance due to the high number of tested hypotheses. This effect triggered a recent discussions about the use of p-values in scientific publications [Wasserstein and Lazar, 2016].

### 5.4. Related work

There are both structural and statistical approaches to extract patterns from time series. Many statistical approaches rely on structures that allow the usage of genetic algorithms. They express the feature pattern for example as a tree [Mierswa and Morik, 2005, Geurts,

2001, Eads et al., 2002]. While doing so, they aim for the best pattern and the most explaining features by alternating and optimizing the used feature mappings. In contrast, FRESH extracts the best fitting of a fixed set of patterns.

As an example for a structured pattern extraction, in [Olszewski, 2001] the authors search for six morphology types: constant, straight, exponential, sinusoidal, triangular, and rectangular phases. Those phases are detected by structure detectors which then output a new time series whose values stand for the identified structure. Based on this structure a domain-independent structural pattern recognition system is utilized to substitute the original time series signal by a known pattern. Due to its fixed patterns, FRESH can be considered to be a structured pattern extractor.

Of course, there are other promising approaches like the combination of nearest neighbor search with Dynamic Time Warping [Wang et al., 2006], which is specialized on considering an ensemble of exactly one dedicated time series type and cannot take meta-information into account. For binary classifications it scales with  $\mathcal{O}(n_t^2 \cdot m_{\text{train}} \cdot m_{\text{test}})$  [Penserini and others, 2006] with  $m_{\text{train}}$  and  $m_{\text{test}}$  being the number of devices in the train and test set, respectively. This approach also has the disadvantage that all data have to be transmitted to a central computing instance.

The extraction algorithm most similar to ours is presented by Fulcher and Jones [2014]. It applies a linear estimator with greedy search and a constant initial model to identify the most important features, which has been considered in this paper as LDA. The evaluation has shown, that FRESH outperforms the approach of Fulcher and Jones [2014]. Also, FRESH provides a more general approach to time series feature extraction, because it is able to extract features for regression tasks and not only for classification.

Despite the applications for time series classification and regression, the feature selection approach of FRESH could be included into kernel methods [Cho and Saul, 2009] and therefore should find broad applicability in the machine learning community.

## 6. Summary and future work

In this work, **FeatuRe** Extraction based on **Scalable Hypothesis** tests (**FRESH**) for time series classification and regression was introduced. It combines well established feature extraction methods with a scalable feature selection based on non-parametric hypothesis tests and the Benjamini-Yekutieli procedure. **FRESH** is highly parallel and suitable for distributed IoT and Industry 4.0 applications like predictive maintenance or process line optimization, because it allows to consider several different time series types per label and additionally takes meta-information into account. The latter has been demonstrated on basis of a steel billets process line optimization of project iPRODICT.

Our evaluation for UCR time series classification tasks has shown that FRESH in combination with a subsequent PCA outperforms all other feature extraction algorithms with respect to scalability and achieved accuracy. On the iPRODICT data set, it was even able to achieve a higher accuracy than a nearest neighbor search under Dynamic Time Warping.

The parallel nature of FRESH with respect to both feature extraction and filtering makes it highly applicable in situations where data is fragmented over a widespread infrastructure and computations cannot be performed on centralized infrastructure. Due to its robustness

and applicability to machine learning problems in the context of IoT and Industry 4.0, we are expecting that FRESH will find widespread application.

## Acknowledgement

This research was funded in part by the German Federal Ministry of Education and Research under grant number 01IS14004 (project iPRODICT).

## References

- Leta McKinney Adler. A Modification of Kendall's Tau for the Case of Arbitrary Ties in Both Rankings. *Journal of the American Statistical Association*, 52(277):33–35, 03 1957.
- Yoav Benjamini and Yoel Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 289–300, 1995.
- Yoav Benjamini and Daniel Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Annals of statistics*, pages 1165–1188, 2001.
- Verónica Bolón-Canedo, Noelia Sánchez-Marroño, and Amparo Alonso-Betanzos. *Feature Selection for High-Dimensional Data*. Artificial Intelligence: Foundations, Theory, and Algorithms. Springer International Publishing, 2015.
- Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The UCR Time Series Classification Archive. [http://www.cs.ucr.edu/~eamonn/time\\_series\\_data/](http://www.cs.ucr.edu/~eamonn/time_series_data/), 07 2015.
- Youngmin Cho and Lawrence K. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems 22*. Curran Associates, Inc., 2009.
- Maximilian Christ, Frank Kienle, and Andreas W. Kempa-Liehr. Time Series Analysis in Industrial Applications. In *Workshop on Extreme Value and Time Series Analysis*, KIT Karlsruhe, 03 2016. doi: 10.13140/RG.2.1.3130.7922.
- Douglas Curran-Everett. Multiple comparisons: philosophies and illustrations. *American Journal of Physiology - Regulatory, Integrative and Comparative Physiology*, 279(1):R1–R8, 07 2000.
- Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 10 2012.
- Damian R. Eads, Daniel Hill, Sean Davis, Simon J. Perkins, Junshui Ma, Reid B. Porter, and James P. Theiler. Genetic Algorithms and Support Vector Machines for Time Series Classification. In Bosacchi et al., editor, *Proc. SPIE 4787, Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation V*, pages 74–85, 12 2002.



- Ramez Elmasri and Jae Young Lee. Implementation options for time-series data. In Opher Etzion, Sushil Jajodia, and Suryanarayana Sripada, editors, *Temporal Databases: Research and Practice*, pages 115–128. Springer Berlin Heidelberg, 1998. URL <http://dx.doi.org/10.1007/BFb0053700>.
- R. A. Fisher. On the Interpretation of  $\chi^2$  from Contingency Tables, and the Calculation of P. *Journal of the Royal Statistical Society*, 85(1):87, 01 1922.
- Ben D. Fulcher and Nick S. Jones. Highly comparative feature-based time-series classification. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):3026, 2014.
- Pierre Geurts. Pattern extraction for time series classification. In *Principles of Data Mining and Knowledge Discovery*, pages 115–127. Springer, 2001.
- Eilam Gross and Ofer Vitells. Trial factors for the look elsewhere effect in high energy physics. *The European Physical Journal C*, 70(1-2):525–530, 2010.
- Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645, 09 2013.
- Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.
- M. Hermann, T. Pentek, and B. Otto. Design Principles for Industrie 4.0 Scenarios. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, page 3928, 01 2016.
- Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.
- Jestinah M Mahachie John, François Van Lishout, Elena S Gusareva, and Kristel Van Steen. A robustness study of parametric and non-parametric tests in model-based multifactor dimensionality reduction for epistasis detection. *BioData mining*, 6(1):1, 2013.
- M. G. Kendall. A New Measure of Rank Correlation. *Biometrika*, 30(1/2):81, 06 1938.
- Kenji Kira and Larry A. Rendell. A Practical Approach to Feature Selection. In *Proceedings of the Ninth International Workshop on Machine Learning, ML92*, pages 249–256. Morgan Kaufmann Publishers Inc., 1992.
- Miron B. Kursa and Witold R. Rudnicki. The all relevant feature selection using random forest. *arXiv preprint arXiv:1106.5112*, 2011.
- Andrew Kusiak and Wenyan Li. The prediction and diagnosis of wind turbine faults. *Renewable Energy*, 36(1):16–23, 01 2011.
- Andreas W. Liehr. *Dissipative Solitons in Reaction Diffusion Systems*, volume 70 of *Springer Series in Synergetics*. Springer, 2013.
- Huan Liu and Rudy Setiono. Some issues on scalable feature selection. *Expert Systems with Applications*, 15(3–4):333 – 339, 1998.

- Frank J. Massey. The Kolmogorov-Smirnov Test for Goodness of Fit. *Journal of the American Statistical Association*, 46(253):68, 03 1951.
- Ingo Mierswa and Katharina Morik. Automatic feature extraction for classifying audio data. *Machine learning*, 58(2-3):127–149, 2005.
- R Keith Mobley. *An introduction to predictive maintenance*. Elsevier Inc., 2 edition, 2002.
- Roland Nilsson, José M. Peña, Johan Björkegren, and Jesper Tegnér. Consistent feature selection for pattern recognition in polynomial time. *The Journal of Machine Learning Research*, 8:589–612, 2007.
- Isadora Nun, Pavlos Protopapas, Brandon Sim, Ming Zhu, Rahul Dave, Nicolas Castro, and Karim Pichara. FATS: Feature Analysis for Time Series. *arXiv preprint arXiv:1506.00010*, 2015.
- Robert T. Olszewski. Generalized feature extraction for structural pattern recognition in time-series data. DTIC Document CMU-CS-01-108, Carnegie Mellon University, 2001.
- Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and others. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.
- L Penserini and others. A comparison of two machine-learning techniques to focus the diagnosis task. *Frontiers in Artificial Intelligence and Applications*, page 265, 2006.
- Predrag Radivojac, Zoran Obradovic, A. Keith Dunker, and Slobodan Vucetic. Feature Selection Filters Based on the Permutation Test. In *Machine Learning: ECML 2004*, pages 334–346. Springer, 2004.
- J. Timmer, C. Gantert, G. Deuschl, and J. Honerkamp. Characteristics of hand tremor time series. *Biological Cybernetics*, 70(1):75–80, 1993.
- Xiaoyue Wang, Abdullah Mueen, Hui Ding, Goce Trajcevski, Peter Scheuermann, and Eamonn Keogh. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 26(2):275–309, 2013.
- Xiaozhe Wang, Kate Smith, and Rob Hyndman. Characteristic-Based Clustering for Time Series Data. *Data Mining and Knowledge Discovery*, 13(3):335–364, 09 2006.
- Ronald L. Wasserstein and Nicole A. Lazar. The ASA’s statement on p-values: context, process, and purpose. *The American Statistician*, 03 2016.
- Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *ICML*, volume 3, pages 856–863, 2003.