Data-Driven Multi-objective Optimization of Industrial End-of-Line Testing Cycles via Wrapper Feature Selection

1st Stefan Gaugel *Bosch Rexroth AG,* Ulm (Germany), E-Mail: stefan.gaugel@boschrexroth.de
2nd Manfred Reichert *Institute of Databases and Information Systems Ulm University* Germany

***Abstract*—Functional end-of-line testing is a powerful but expensive approach for industrial quality assurance. A major cost driver of end-of-line testing is the often overlong and overcomplex design of industrial testing cycles. To tackle this challenge, the paper proposes a fully-automated approach based on machine learning for the optimization of end-of-line testing cycles. As each testing cycle comprises a multivariate and multi-phased time series, the optimization can be interpreted as a form of feature selection that aims to select the minimal subset of relevant and non-redundant testing sensors and cycle phases. We define the problem as a multi-objective optimization task balancing the goals of maximizing the fault detection accuracy while minimizing the testing cycle complexity. The general idea of the optimization approach is inspired by the iterative concept of wrapper feature selection. Our approach was validated in an experiment by applying it to real-world end-of-line testing data of hydraulic pumps. The results have shown that the approach can be successfully used to find an improved testing cycle design and, especially, Backward Elimination performs well as solution search algorithm.**

I. Introduction

A. Problem Statement

In recent years, the utilization of artificial intelligence and other data-centric methods in manufacturing have been continuously growing. In modern production plants, a vast amount of sensor data is collected each day, increasing the need for methods to create value from the collected data [1]. Sensor data are usually available in a time series format. Consequently, typical tasks from the time series domain, such as classification, segmentation, and forecasting, have received a high level of attention in recent industrial research. The most prevalent industrial applications of data-driven methods are condition monitoring, predictive maintenance, and human activity recognition [2]. In this article, however, we focus on a different task with very little prior research: the detection and elimination of redundant or irrelevant segments in industrial time series data.

The industrial problem tackled in this paper deals with the optimization of testing cycles in End-of-Line-Testing (EoLT). In EoLT, before its delivery a product is mounted at a testbench and forced through a testing cycle to examine its quality and to detect product faults. Testing cycles have a multi-phased nature, meaning that each cycle follows a predefined order of clearly distinguishable testing cycle phases with abrupt changepoints between them [3]. When designing testing cycles, engineers have three conflicting goals. First, they choose a testing cycle design which ensures sufficient data is gathered to accurately assess the tested product as good or faulty (high faulty detection accuracy); Second, the cycle time should be minimal; and third, the number of required sensors should be minimal. Many plants that manufacture complex technical products perform a 100% quality examination, meaning that each manufactured product is tested before delivery. Consequently, a minimization of testing cycle time is a valuable cost leverage for the entire plant. Additionally, an high number of sensors directly translates into high effort for EoLT preparation and testbench maintenance. Finding the right balance between high fault detection accuracy and minimal testing cycle complexity is a challenging task for engineers.

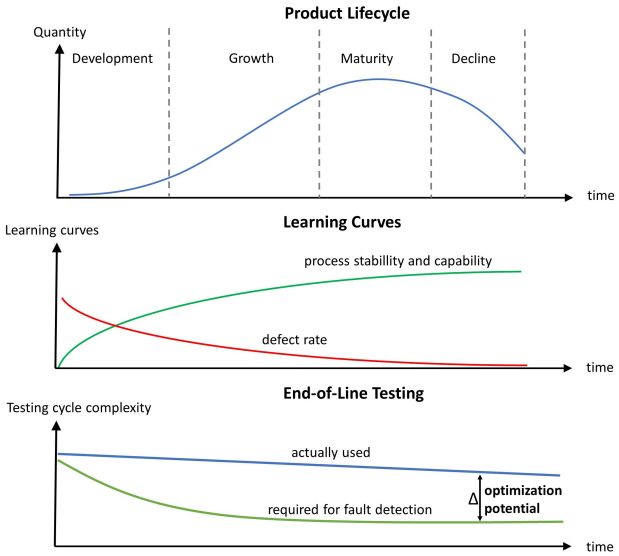

Fig. 1: Development of process stability and defect rate over the lifecycle of a product

In the initial stages of a product life cycle, process stability is usually still low. In turn, this causes high defect rates compared to later stages. To detect the numerous defects in the initial stages and to create customer trust in product quality, engineers typically focus heavily on achieving high fault detection accuracies in the initial testing cycle designs (Figure 1). As a result, sometimes overly-long testing cycles

with multiple redundancies are defined in the early product life [4]. Later life cycle stages with higher process stability then allow for testing cycle optimization by eliminating redundant and irrelevant testing phases and sensors. Optimization offers high potential, as EoLT often causes costs as high as 6% of a products revenue [5]. However, the optimization is often not realized by manufacturers due to time and capacity issues as well as the fear of decreasing quality control.

B. Own Contribution

Designing EoLT cycles is traditionally a very knowledge-intensive process which usually requires expertism from the fields of mechanical engineering, electrical engineering, and production line design [6]. Recently, however, data-driven methods have started to find their way into the field, focusing on topics like supervised classification [7], anomaly detection [8], and operational state segmentation [9]. Nevertheless, no previous contribution covered the topic of EoLT cycle optimization. To fill that gap, our work proposes the first fully-automated and data-driven approach for testing cycle optimization. We design the approach in a domain-independent manner to ensure its applicability to all sorts of testing cycle optimization problems. For the approach, examining a tested products as healthy or faulty based on EoLT sensor data is transformed into a supervised Machine Learning (ML) classification task. The feature extraction is performed in isolation for each sensor and testing state, as past research has shown that state-specific feature extraction performs best for the specific multivariate and multi-phased character of the time series found in EoLT [3]. Testing cycle optimization, in turn, is defined as the task of finding the minimal subset of sensors and cycle phases that keeps the fault detection accuracy at the highest possible level. Only features extracted from the time series frames for which the selected sensor subset and the selected cycle-phase subset intersect are allowed to be included in the fault detection ML model (see Figure 2).

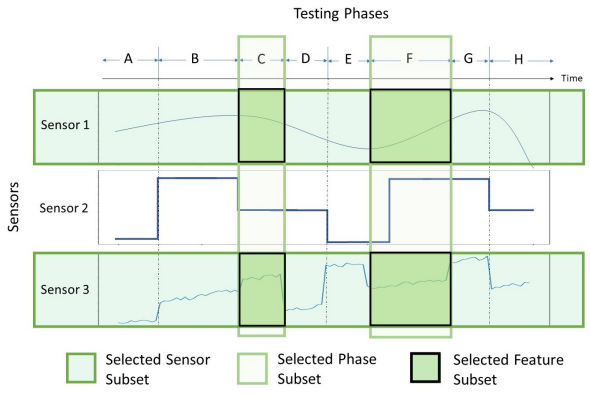

Fig. 2: Testing Cycle Optimization via Feature Selection in Multivariate Multi-Phased Time Series

Our approach is based on the principle of wrapper feature selection. In wrapper methods, different feature combinations are generated by a predefined algorithm and evaluated via an objective function (see Figure 4). In the case of EoLT, testing cycle optimization is defined as a multi-objective optimization problem that tries to find the ideal data subset that simultaneously maximizes the fault detection precision and recall of a machine learning classifier, and minimizes the cycle time and number of sensors of the subset used to train the classifier. A solution candidate for the optimization problem is represented by an encoding, specifying which phases and sensors are available for the feature extraction process before training the classifier. For the solution search, common feature selection search algorithms are examined, including traditional algorithms such as Forward Selection, and Backward Elimination, as well as metaheuristics such as Genetic Algorithm, and Simulated Annealing [10].

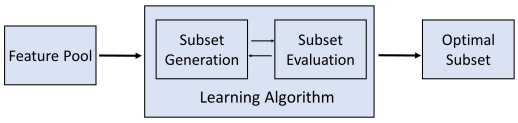

Fig. 3: Concept of wrapper feature selection

The approach is validated in the experimental section on a public dataset representing the EoLT data of hydraulic pumps [9]. The weights in the objective function are set in three different ways to simulate different potential cost structures of manufacturers. As our approach represents a special type of wrapper feature selection, two baselines representing a filter feature selection method and an embedded feature selection method are implemented for comparison purposes.

Altogether, the main contributions of this article are two-fold, i.e., (1) proposal of the first generally-applicable approach for automated data-driven testing cycle optimization and (2) application and validation of the approach to a real-world hydraulic pump EoLT use case. Additionally, our work provides evidence that Wrapper feature selection is able to successfully detect irrelevant or redundant elements in multivariate multi-phased time series, which is something that has not been investigated before. Finally, we also provide a comparison of the performance of different heuristic search algorithms in the domain of test cycle optimization.

The remainder of this work is structured as follows: Section 2 presents the results of a literature review. Section 3 introduces the approach, the experimental setup, and the used dataset in detail. Section 4 summarizes and discusses the experimental results. Section 5 concludes the article and proposes directions for future research.

II. Related Works

As our approach is based on the concept of wrapper-based feature selection, we worked through past research on feature selection in the time series domain and on data-driven approaches for EoLT. Therefore, both the methodical and the application side of our approach are covered.

A. Feature Selection in the Time Series Domain

In general, feature selection is a widely-studied topic in the context of Machine Learning [11],[12]. The goal

of feature selection is to remove irrelevant or redundant elements from the feature pool. Usually, three different kinds of selection approaches are distinguished: filter methods, wrapper methods, and embedded methods [13]. Filter methods are most commonly used. They rely on data characteristics and use statistical metrics to create an importance ranking of the total feature pool, before selecting the best-scoring ones. Wrapper methods are iterative approaches which use a predefined classifier and search algorithm: In each iteration, the search algorithm selects a feature subset. The classifier is then trained on the selected feature subset and the performance is evaluated, before a new feature subset is selected. The best-scoring feature subset over various iterations constitutes the final feature selection. In embedded methods, the feature selection process is embedded in the classifier learning phase. For some trained classifiers (Regression-based, Tree-based), their internal weights can be accessed, interpreted, and used to select the features of highest importance. As all three methods have drawbacks, hybrid methods combining concepts from the different groups have been designed as well [14].

As most time series data are by nature high-dimensional, feature selection is of special importance and difficulty in the time series domain. Most prevalent for feature selection in time series are (1) filter-methods and (2) hybrid methods combining filter and wrapper concepts. Especially the detection and elimination of variables with high redundancy or low relevance has achieved high attention. [15] used a filter-method to identify the subset of relevant and non-redundant variables in multivariate time series classification. Their approach calculated both the mutual information between two time series variables (redundancy metric) and between a time series variable and the class variable (relevance metric). Transferred to our use case, this approach would aim to identify the minimal subset of required sensors for fault detection. In a similar manner, [16] proposed a filter-based framework to detect the most important sensors in manufacturing for fault detection using a minimum redundancy maximum relevance concept. In the area of industrial process monitoring, [17] presented a principal component analysis-based approach for variable subselection in multivariate time series data using redundancy and relevance metrics. [18] presented a combination of a minimum redundancy maximum relevance-based filter method and a recursive feature elimination wrapper method for time series feature selection. [19] proposed a new framework for a combined wrapper-filter approach based on Forward Selection in order to forecast solar radiation time series data. In the area of stock price forecasting, [20] introduced a combination of wrapper and filter feature selection based on a Genetic Algorithm.

For this article, the approach we design is based on the wrapper concept, as filter approaches do not work well for multi-label classifications problem with numerical features. An overview of wrapper feature selection approaches can be found in [21]. In the time series domain, pure wrapper methods for feature selection are less frequently encountered compared to the other methods. One example is [22], which shows the use a wrapper-based feature selection approach based on multi-objective particle swarm optimization and an ML classifier to forecast the crude oil price. However, no work has proposed a wrapper method directly applicable to the multi-objective testing cycle optimization problem described in this article, in which both a sensor subset and a testing phase subset have to be selected. To the best of our knowledge, the specific multi-phased and multivariate characteristics of testing cycles and their implications on the detection and elimination of irrelevance and redundancy have not been addressed before.

B. Data-driven approaches in End-of-Line Testing

In recent years, numerous studies have been conducted to explore the application of ML and other data-driven methods in the setting of EoLT. [23] proposed an approach to the EoLT problem of replacing a physical sensor with a virtual one whose values are created by deep learning predictions. [24] proposed a machine learning approach to classify the inertial load of electromechanical actuators in EoLT. [25] proposed an EoLT-tailored data analytics framework that supports operators in fault diagnosis and quality assurance. Additionally, [26] published a comparative study of different data-driven fault detection models in an EoLT setting. [8] used a one-class ML classifier to detect product faults as model anomalies in EoLT. Finally, [27] proposed an approach for the automated calculation of test limits by robust stochastic state-space models. However, none of the aforementioned works focused on the topic of testing cycle optimization in EoLT.

III. Proposed Methodology

A. Machine Learning in End-of-Line Testing Settings

The optimization approach we propose is based on algorithms from the ML domain. Traditional EoLT, however, is typically conducted using a rule-based system, wherein measured sensor values in different testing states are compared to predefined test limits. Exceeding these limits at any point is equivalent to being classified as faulty. Additionally, visual inspection of characteristic curves may be performed by a human operator. As traditional rule-based systems necessitate human intervention for any change, they are less suitable for automated, data-driven optimization compared to ML systems. Thus, it becomes necessary to transform fault detection in EoLT into a supervised ML problem.

For ML, a dataset is required in which each sample represents the entire EoLT cycle data of one product, meaning all samples are multivariate and multi-phased time series. As in industrial EoLT only a limited share of the total information is relevant for the distinction of healthy and faulty pumps, ML-based quality classification is highly challenging. Extracting and selecting the appropriate features constitutes a key factor in achieving accurate classification. Past research has demonstrated that for good classifier performance, the multi-phased data must be annotated with two types of labels [3], namely state labels and class labels. First, the sample-specific class label (or quality label) specifies the health status of a

product (e.g., 0 for healthy product, 1-5 for different product fault types). The class label corresponds to the label that the ML classifier learns to predict. Second, state labels have to be available for each time stamp within all samples. They serve as auxiliary semantic annotation for the time series specifying the operational state of the product at each timestamp (see Figure 4).

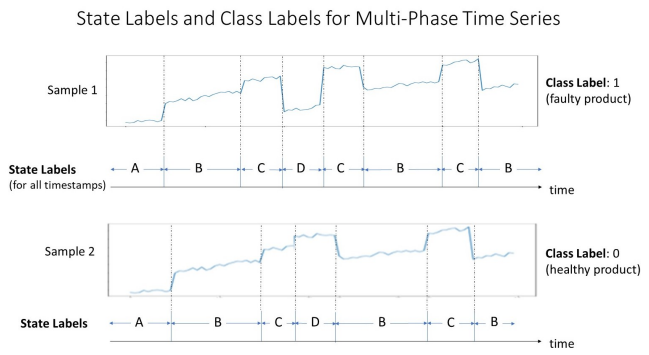

Fig. 4: Two types of labels in EoLT (taken out of [3])

The state labeling can either be done offline after testing by domain experts or online via automated flagging of the testbench during testing. The existence of state labels allows for the isolated extraction of state-specific features than can significantly improve classification accuracy. After completing the local feature extraction process for each state and sensor, it becomes necessary to determine which classifier performs best for the resulting feature pool. Usually, tree-based ensemble methods are best suited for this kind of tasks, as they can imitate the rule-based fault detection system found in EoLT best [3]. After choosing a classifier, its performance on the total available feature pool serves as baseline for the subsequent testing cycle optimization.

B. Multi-objective Optimization via Wrapper Feature Selection

The goal of our testing cycle optimization approach is to find the subset of sensors and testing cycle phases that best fulfills the goal of achieving high fault detection accuracy with minimal testing cycle complexity. Our approach builds on the concept of wrapper feature selection and has a heuristic character. In general, heuristic optimization frameworks consist of three components: solution space, solution candidate evaluation, and solution search strategy. In the following, we give insights into the design of the three components in our approach.

1. Solution Space

The solution space describes the set of all potential solution candidates for a given problem. In wrapper feature selection, it includes the total feature pool and all potential subsets of it. In our specific case, the total feature pool is equivalent to including all available sensors and cycle phases. A solution candidate represents the union of a sensor subset and a phase subset. Consequently, the solution space describes all potential combinations of sensor subsets and phase subsets, ranging from not including any sensor or cycle phase to including all sensors and all cycle phases. A solution candidate is represented by a binary-encoded list of length L with L being equal to the summed-up number of sensors and cycle phases in the original testing cycle design. The binary value found at a specific index indicates whether the respective sensor and phase is included (=1) or not (=0) for a given solution candidate (see Figure 5). Thus, the cardinality of the solution space is given by 2^L.

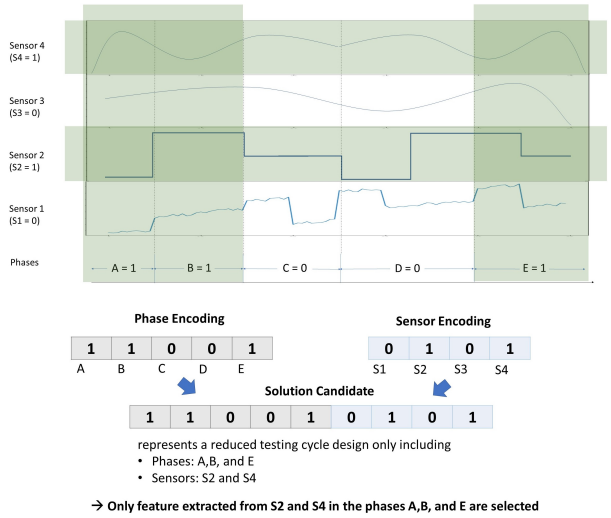

Fig. 5: Solution Candidate Encoding and Implications on Feature Selection

2. Solution Candidate Evaluation

To make different solution candidates comparable, we need a procedure for quantitatively evaluating of each candidate. As testing cycle optimization includes three conflicting goals i.e., maximizing fault detection accuracy, minimizing number of sensors, and minimizing cycle time, with varying importance in different scenarios, we use a weighted linear combination of metrics that represent the different objectives. The weights can be adjusted arbitrarily for different settings.

Goal 1: High Fault Detection Accuracy

Based on the encoding of the solution candidate, only features from the included sensors and cycle phases may be extracted. The details of the state-based feature extraction process may vary. In EoLT, usually the extraction of sensor-wise univariate descriptive statistics (mean, standard deviation, minimum, maximum, etc.) for each state is sufficient. However, if necessary, more complex feature extraction operations may be performed. The resulting feature subset is then used to train the classifier and to evaluate its performance. In turn, the available data are split into training and testing data via stratified sampling to ensure that faulty samples are identically distributed across the training and test set. After using the trained classifier to predict the classes of the test data, the resulting confusion matrix is used to calculate two metrics: the fault detection precision FDP (1) and the healthiness detection precision HDP (2). While the FDP metric describes the percentage of faulty pumps detected by the model, the HDP metric represents the

percentage of healthy pumps predicted correctly. A high FDP expresses that almost all faults get discovered; a high HDP means that not many healthy pumps are wrongfully assumed to be faulty.

$$FDP = \frac{faulty\ products\ predicted\ correctly}{total\ number\ of\ faulty\ products} \quad (1)$$

$$HDP = \frac{healthy\ products\ predicted\ correctly}{total\ number\ of\ healthy\ products} \quad (2)$$

One example for the calculation of FDP and HDP can be found in Figure 6. In case of a multiclass classification problem (different fault types having their own distinct label), there are two options for calculating the FDP. First, the multiclass confusion matrix can be transformed to a binary one by setting the classes of all fault types to be identical. This option follows the assumption that confusing two fault types with each other does not have direct negative implications in EoLT, and only the fact that the defect was detected matters. The second option would be to calculate the FDP as the percentage of all faulty pumps whose fault type was predicted correctly. To obtain more robust results, the calculation of FDP and HDP can be repeated multiple times (e.g., through stratified cross-validation) and averaged at the end. Both FDP and HDP are included in the objective function as separately weighted components. Usually, the weight of FDP is set significantly higher compared to the HDP weight, as the non-detection of faults has more severe consequences compared to their overdetection. Figure 6 contains an exemplary calculation for a given solution candidate.

Goal 2: Minimal Cycle Time

Depending on which cycle phases are excluded in a solution candidate compared to the original cycle design, it has to be checked what extent of time savings can be realized. Before the optimization, the average duration of each phase over all samples has to be calculated. This allows adding up the expected time savings of all phases with zero-encoding of a specific solution candidate. The final metric found in the objective function corresponds to the rate of realised time savings RTS (3). An example for the calculation is displayed in Figure 6.

$$RTS = \frac{realised\ time\ savings}{total\ cycle\ time\ with\ all\ phases\ included} \quad (3)$$

Goal 3: Minimal number of sensors

For each solution candidate it is evaluated how many sensors are excluded in the proposed testing cycle design compared to the original design. This number corresponds to the sensor reduction number. The relative reduction of sensor number RSN compared to the baseline, i.e. the number of sensors in the original testing cycle design, serves as the respective metric in the objective function (see Figure 6 for an example). The formula can be found in (4).

$$RSN = \frac{number\ of\ excluded\ sensors}{total\ number\ of\ available\ sensors} \quad (4)$$

As multi-objective optimization problem are often very difficult to solve, we decided to use the weighted sum method to transform the problem into a single-objective problem. The weighted sum method is a popular methodology to solve multi-objective problems and constitutes of combining all objectives into one singe scalar. This is done by multiplying each objective with a self-determined objective-specific weight and summing up the respective products. Transferred to our problem, the general form of the weighted sum objective function f used to evaluate each solution candidate c can be found in (5). Please note that w_i with i=1,...,4 denotes the respective weight of the goal metrics (1) - (4). The weights can be set arbitrarily to enable setting varying priorities in the optimization. Please note that the higher a respective weight is set, the more emphasis is put on the associated goal during the optimization.

$$f(c) = w_1 FDP(c) + w_2 HDP(c) + w_3 RTS(c) + w_4 RSN(c) \quad (5)$$

Finding the solution candidate that maximizes the function is set as the objective of the optimization approach. The objective function has an ordinal scale, i.e. its value can be used for comparing and ranking the different solution candidates, but has no meaning by itself.

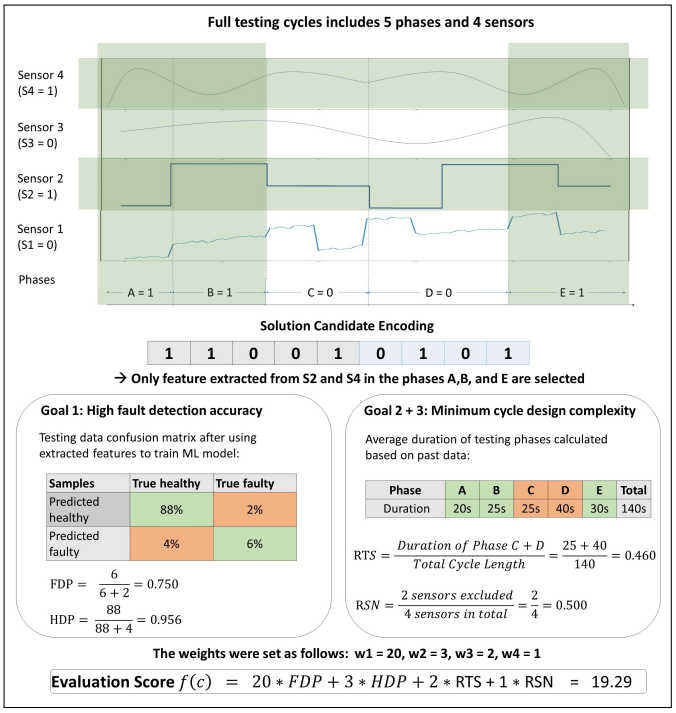

Fig. 6: Example for multi-objective evaluation of a solution candidate

3. Solution Candidate Search

The concept of wrapper feature selection is based on iteratively generating new feasible solution candidates (feature subsets) and evaluating them until a certain stopping criterion becomes fulfilled. Figure 7 depicts the general concept of wrapper feature selection as well as how we adjusted it to create our testing cycle optimization approach.

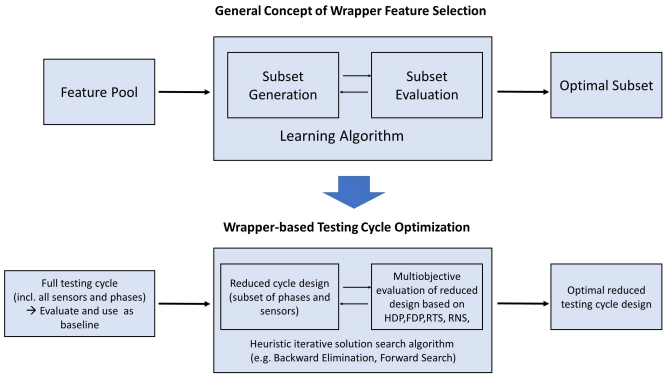

Fig. 7: Testing cycle optimization as a form of wrapper-based feature selection

The generation of new solution candidates corresponds to a search through the solution space. There are various algorithms that can be used for such a solution space search. For our testing cycle optimization setting, we recommend using Backward Elimination. Backward Elimination is a feature selection algorithm that starts with the total feature pool and then eliminates features one by one until the model performance can no longer be improved. In each iteration, all features are evaluated for removal and the feature whose elimination is most beneficial to the total model performance is removed. Transferred to our approach, this means we start with the full testing cycle including all phases and all sensors. Then we remove, in each iteration, the sensor or cycle phase whose elimination leads to the highest immediate gain in the objective function (see Figure 8 for a depiction of the concept). When the objective function can no longer be improved by removing one more element, the algorithm terminates. Possible alternatives to Backward Elimination are Forward Selection algorithms or metaheuristics (see Section IV-B).

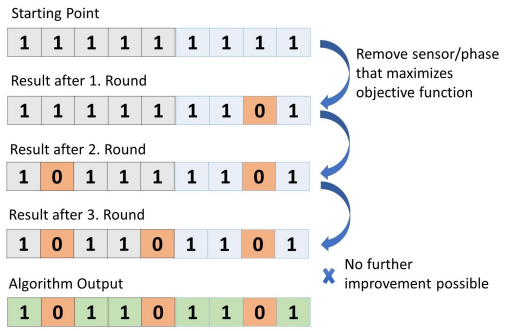

Fig. 8: Visualization of Backward Elimination concept

In general, we recommend the following prerequisites to be fulfilled for our approach to be applicable:

1) An ML classifier is able to achieve excellent fault detection results for the original testing cycle.
2) Sensors and phases included in the original cycle design can be removed independently without affecting other elements of the testing process.
3) The available samples have a sample-specific class label and timestamp-specific state labels.
4) Enough samples are available to be able to apply a data-driven approach (at least 100). Additionally, enough labeled fault data exist (more than 5 for each fault type). The numbers are a rough estimate based on experience.

IV. Experiments

A. Dataset

To validate our approach, we use the publicly available Hydraulic-End-of-Line-Testing (EoLT) dataset. The dataset includes the EoLT data of hydraulic pumps measured by sensors of hydraulic testbenches in a German production plant. The data are stored and accessible on Git via https://github.com/boschresearch/Hydraulic-EoL-Testing/. The dataset was already used in some of our earlier works covering time series segmentation [9] and transfer learning [28]. For our experiments, we use the total of 202 Generation B pump samples. One sample is equivalent to one multivariate time series, representing the entire testing cycle of a unique manufactured pump. Nine sensors were used during EoLT. Consequently, each sample contains nine time-dependent variables representing the sensors. As per request of the industrial partner, the sensor names and scales were removed and the sensor values were normalized (z-score standardization). 136 of the 202 samples are labeled healthy, whereas 66 are labeled faulty. The 66 faulty samples contain 11 different fault types. Each fault type is found in 6 samples to ensure that enough data for each type exists. Different fault types have very different fault signals, making it difficult to assign them to one shared fault class for machine learning. Consequently, each fault type gets a distinct class label assigned (1-11 for the different fault types). The 0-label indicates a healthy pump. Furthermore, all samples are state-labeled, i.e., each timestamp within a sample has an associated state label indicating the current operational state of the pump. The 42 available integer-encoded state labels structure the time series into multiple subsequences and help extracting local state-specific features.

B. Preparation of Data and Experiment

Each sample of the Hydraulic Pump EoLT data, which we use for our experiments, includes nine different sensors and up to 42 different state labels. One whole sample of the dataset (including the labeling) is exemplarily depicted in Figure 9. Of the nine sensors, three are necessary to control the EoLT process and pump flow, leaving six sensors as removal candidates. Additionally, in the context of testing cycle optimization, one operational state cannot always be set identical to one testing phase. Different states might have a mutual existential dependency and can not be removed independently. For example, a state representing a static testing point is usually preceded by a dynamic state representing the transition of the pump from a neutral state to the specified testing point. Therefore, if such existential dependencies exist, it becomes necessary to assign the 42 different states to a lower number of 15 cycle phases, where each phase contains multiple operational states. The phases were defined with the highest possible granularity that still enables their independent removal. Table I shows the

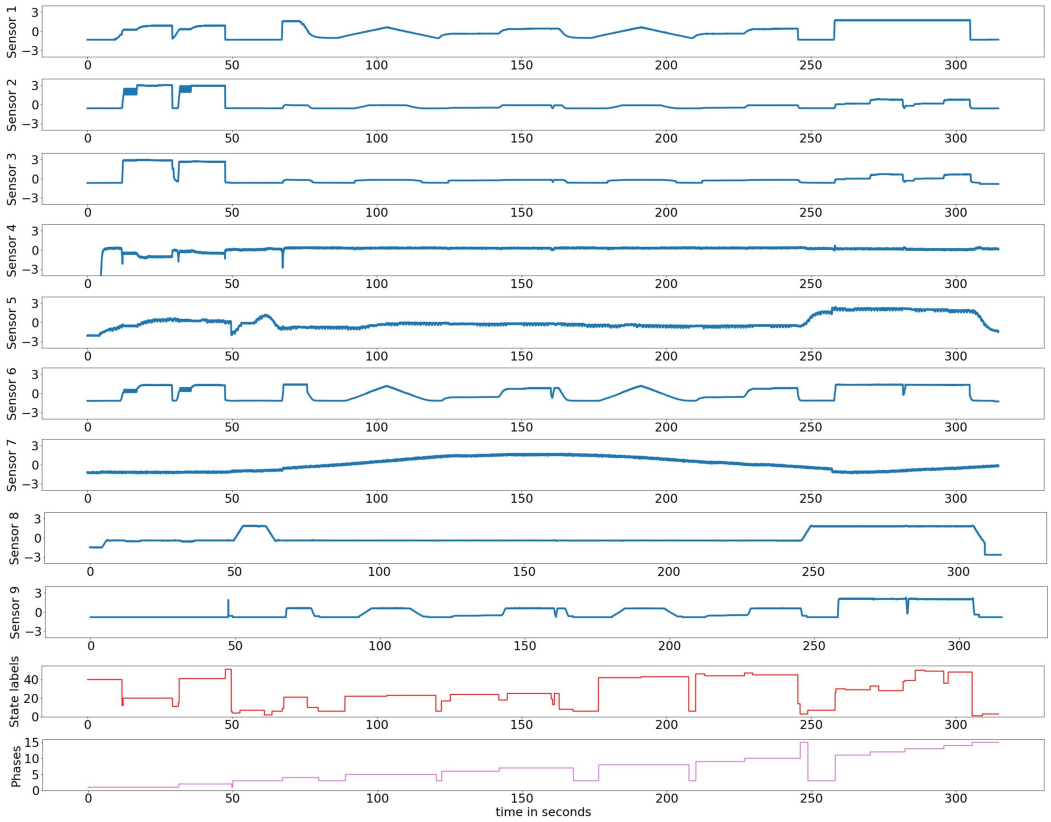

Fig. 9: Full visualization of one testing cycle sample including state labels and associated testing cycle phases

assignment of the state labels to the defined cycle phases. Therefore, a solution candidate encoding in our setting has the length of 21, constituted by the six sensor removal candidates and the 15 phases (see Figure 10). We calculated

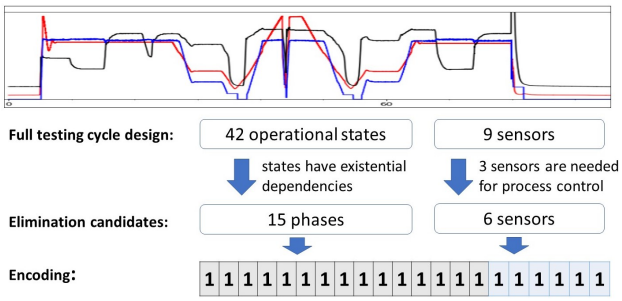

Fig. 10: Encoding of full EoLT hydraulic pump testing cycle

the average duration of each phase over all samples to quantify the time-saving potential benefit of removing a phase from the cycle. Before the start of the experiment, we used the solution candidate representing the existing testing cycle (all 21 elements equal 1) to find the best ML strategy (classifier, extracted features, preprocessing steps). Based on the results, we decided to use an Random Forest classifier with the specifications found in Table III.

Cycle Phase	Associated States
1	11, 12, 20, 40
2	15, 41, 51
3	2, 4, 7
4	9, 10, 21
5	22, 23
6	17, 24
7	8, 13, 18, 19, 25
8	42, 43
9	44, 46
10	14, 45, 47
11	26, 29, 30
12	28, 33, 38
13	39, 49, 50
14	36, 38
15	1, 3

TABLE I: Testing Cycle Phases and associated operation states

C. Experimental Set-Up

We used three different weight sets for the objective functions in our experiments representing three different production plant cost structures. The first objective function (OF1) balances the different objectives (w_1=8, w_2=2, w_3=1, w_4=1.5), the second one (OF2) puts a maximum emphasis on keeping a very high fault detection accuracy (w_1=30, w_2=10, w_3=1, w_4=1.5), the third one (OF3) puts maximum emphasis on reducing testing cycle complexity, even at the cost of small fault detection accuracy decreases (w_1=3, w_2=1, w_3=1.5, w_4=1.5). It has to be mentioned that the absolute value of the

Search Algorithm	Principle	Parameters
Backward Elimination (BackElim)	starts with all predictors and eliminate them iteratively. At each subsequent iteration, one further remaining predictor is eliminated based on performance criteria.	-
Forward Selection (ForwSe)	starts with one predictor and adds more iteratively. At each subsequent iteration, the best of the remaining original predictors are added based on performance criteria.	-
Random Search (RandSe)	randomly generate solution candidates by including each phase or sensor with 50% chance	-
Genetic Algorithm (GenAlg)	metaheuristic for solving search problems based on natural selection, relying on biologically inspired operations like mutation, crossover and selection	num_generations = 20,sol_per_pop = 8, num_parents_mating = 4, crossover_type =”single_point”, mutation$_{p}ercent_{g}enes = 30$
Simulated Annealing (SimAnn)	metaheuristic to help searching in large spaces, aims to find approximate global optimum, the concept is based on annealing in metallurgy	start_temperature = 10, numb_epochs=50, inital_sol = encoding of full cycle

TABLE II: Overview of heuristic solution search algorithms used in the experiment

ML Strategy Component	Selected Specification used in Experiments
Classifier	Random Forest (n_estimators=100, max features=200, max depth=80)
Train/Test-Split	Stratified sampling, 2/3 train data, 1/3 test data
Class Imbalance Handling	Oversampling (3 times) of fault class samples in training set
Extracted Features	Mean and standard deviation for each state-sensor combination

TABLE III: Details about Machine Learning (ML)-Strategy

objective function itself does not possess any meaning, but only enables the comparison and ordinal ranking of various solution candidates.

For the solution space search, we evaluated the performance of five different algorithms (Table II). As most of them include stochasticity, we ran them five times for each objective function and reported the mean over the five runs. As our approach is founded on the concept of wrapper feature selection, we implemented two baselines representing the other feature selection concepts, i.e., filter-based feature selection and embedded-feature selection. For filter-based feature selection, we eliminated all features that had both a correlation to another feature higher than 0.7 and a point-biserial correlation with the binary class label that was lower compared to that of the correlated feature. For embedded-feature selection, the 60 most important features based on the embedded Random Forest feature importance scores were selected after training the classifier. For both baselines, it was then evaluated from which phases and sensors the selected features were extracted. Consequently, phases and sensors from which no features were extracted were cut from the testing cycle design. Based on the results, the respective optimization function score was calculated.

V. Results

The results of the experiments are displayed in Table IV. As the three objective functions differ significantly in the selected weights, the results can only be compared within one table line, but not across lines. We compare the wrapper approach (using the five solution search algorithms found in Table II) with two reference scores (global optimum and full cycle) and the two baseline approaches. A cycle optimization is successfully reached when the full cycle reference score is exceeded. We can see that the wrapper approach using Backward Elimination is able to come up with an improved solution compared to the full cycle reference score for all objective functions. In addition to Backward Elimination, only simulation annealing detected solutions achieving a cycle optimization for all three objective functions. Notably, both algorithms share the principle that the solution search starts with the full cycle, before removing phases and sensors one by one in subsequent iterations. The results provide evidence that this principle seems to be the most suitable for achieving a testing cycle optimization.

For Forward Selection and Genetic Algorithm, the results varied significantly for the different objective functions. While Forward Selection found improved testing cycles for the second and third objective functions, it scored very low for the first one. The low score can be explained by an overly early termination of the algorithm. In particular, the risk of early termination is present in very early stages of the algorithm when the included phases and sensors are so few that no single add-on can improve the total score. The Genetic Algorithm outperformed the full cycle reference only in the third objective function, while falling short in the first two. It seems to fail to provide an effective search logic

Objective Function	Reference Scores		Multi-objective Wrapper Approach					Baselines	
	Global Optimum	Full Cycle	BackElim	ForwSe	RandSe	GenAlg	SimAnn	Filter	Embedded
OF1	11.30	10	11.27	6.93	8.71	9.81	11.24	7.2	9.89
OF2	41.20	40	41.01	40.94	32.67	37.05	40.78	29.31	38.79
OF3	5.44	4	5.43	4.88	4.5	5.13	4.92	4.19	3.82

TABLE IV: Objective function values of the different cycle optimization approaches for the objective functions (OF1, OF2, and OF3) compared to reference scores

for testing cycle optimization problems. Random search, as expected, constantly scored worst of the solution search algorithms, as no underlying search logic was used for the solution search. However, in the third setting, despite relying on the random generation of solution candidates, it still could exceed the full cycle reference. The third setting put high emphasis on time savings and less on high accuracy, leading to a high number of solution candidates outperforming the full cycle solution compared to the other two objective functions. Therefore, the chances of Random Search to detect one of these solutions were significantly higher compared to the other two scenarios. No algorithm was able to detect the global maximum in any of the three settings. However, Backward Elimination detected a close-to-optimal solution for all objective functions. The results, therefore, confirm our recommendation of using Backward Elimination as the standard solution search algorithm. The baselines performed consistently poor, meaning filter-based feature selection and embedded feature selection seem not suitable for testing cycle optimization.

Figure 11 exemplarily shows the score (including the metrics to determine it) and encoding of three solution candidates (full cycle, global optimum, Backward Elimination)in the scenario of objective function OF2. The score of the full-cycle reference is based on perfect model accuracy, zero time savings, and zero sensor reductions. The global optimum is successful in selecting the features in a way that the perfect model accuracy is almost maintained, while the achieved time savings and sensor reductions are maximized. The global optimum represents the relevant, non-redundant subset of features necessary to build an excellent model. The solution candidates detected by Backward Elimination and Simulated Annealing are very close to the global maximum and achieve considerable cycle time savings (20% for Backward Elimination and 9% for Simulated Annealing) and a sensor number reduction by three. Therefore, while both approaches were able to detect the optimal sensor subset, they did not realize the full time-saving potential.

The presented cycle optimization approach is conceptualized as a heuristic, meaning that a low runtime is considered more important than the optimality of the result. Table V shows the measured runtime of the different search algorithms compared to the calculated runtime of a brute-force algorithm evaluating all 2^{21} possible solution candidates. Clearly, the runtime of the approach for the different search algorithms is unproblematic in practice, ranging from 400s to 1440s. Especially when using Simulated Annealing and Backward

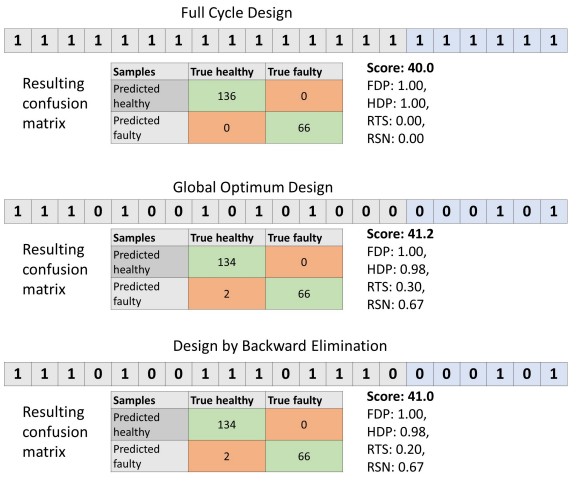

Fig. 11: Solution encodings and respective evaluation scores (based on objective function OF2)

Elimination, close-to-optimal solutions can be found within a very reasonable amount of time compared to a brute-force algorithm guaranteeing optimality.

Algorithm	Runtime
Backward Elimination	1440s
Forward Selection	1240s
Random Search	700s
Simulated Annealing	400s
Genetic Algorithm	1280s
Brute-Force	14680064s (est.)

TABLE V: Runtime comparison of the different approaches

In the following, we summarize the three main findings:

1) The proposed approach was able to detect an improved testing cycle design when applied to a hydraulic pump EoLT dataset. Therefore, we have shown that wrapper feature selection can be used to eliminate redundant or irrelevant elements from multi-phased multivariate time series.
2) Search algorithms which start including all features before gradually eliminating elements (Backward Elimination, Simulated Annealing) outperformed other search algorithms.
3) While the proposed approach was never able to detect the optimal solution, it was able to detect close-to-optimal testing cycle designs within a short amount of time. This finding is in line with the heuristic character of wrapper feature selection.

VI. Summary and Outlook

This article proposed a data-driven heuristic framework for EoLT cycle optimization. The optimization is approached as a form of wrapper feature selection in the multivariate multi-phased time series domain. We use a multi-metric objective function that includes the conflicting objectives of testing cycle optimization, namely both the maximization of the product quality assessment accuracy and the minimization of testing complexity (cycle length, number of required sensors). The presented approach requires the testing cycle time series data to have two types of labels: First, one sample-specific class label providing information about the pump quality and second, timestamp-specific state labels that segment the testing cycle and enable the extraction of state-specific features. The problem of fault detection is transformed into a ML problem which serves as the foundation of the feature subset selection algorithm used in the subsequent optimization algorithm. A solution candidate is encoded into a binary list that provides information about which cycle phases and sensors are included for the respective feature extraction process. For the solution generation, various solution space search algorithms can be used (e.g., metaheuristics). We recommend using Backward Elimination. We validated the approach on a dataset representing the EoLT data of hydraulic pumps. The experiments have shown that the approach was able to find a close-to-optimal solution in a reasonable amount of time for three different objective functions representing different plant cost structures. In particular, BE and SA with a full testing cycle as starting points showed good results as solution search algorithms.

Limitations of the study include the high prerequisites for the designed approach to be applicable, the lack of recommendations on how to weight the different metrics in the objective function, and the application of the approach in only one setting. Additionally, the results on which search algorithms perform best are only based on one use case, which is not enough to establish the general validity of the findings. Further research should validate our approach on additional EoLT datasets and perform an empirical in-depth search about which search algorithms are more or less suitable for different settings. Moreover, more automated data-driven approaches for testing cycle optimization should be designed and the results compared to the ones of this paper.

References

[1] K. Kammerer, R. Pryss, B. Hoppenstedt, K. Sommer, and M. Reichert, "Process-driven and flow-based processing of industrial sensor data," *Sensors*, vol. 20, no. 18, 2020, ISSN: 1424-8220. DOI: 10 . 3390 / s20185245. [Online]. Available: https://www.mdpi.com/1424-8220/20/18/5245.

[2] B. Hoppenstedt, R. Pryss, B. Stelzer, *et al.*, "Techniques and emerging trends for state of the art equipment maintenance systems—a bibliometric analysis," *Applied Sciences*, vol. 8, no. 6, 2018, ISSN: 2076-3417. DOI: 10.3390/app8060916. [Online]. Available: https://www.mdpi.com/2076-3417/8/6/916.

[3] S. Gaugel, B. Wu, A. Anand, and M. Reichert, "Supervised time series segmentation as enabler of multi-phased time series classification: A study on hydraulic end-of-line testing," *Preprint (academia.edu)*, 2023.

[4] T. Schaefer, "Fallstudie end-of-line pruefstand - hochautomatisierte 100 prozent-prüfung am beispiel der massenfertigung von proportionalventilen," Feb. 2021.

[5] S. Teli and A. Murumkar, "Cost of quality for automobile industry: A review," Sep. 2018.

[6] W. Klippel, "End-of-line testing," in *Assembly Line*, W. Grzechca, Ed., Rijeka: IntechOpen, 2011, ch. 10. DOI: 10.5772/21037. [Online]. Available: https://doi.org/10.5772/21037.

[7] W. Shang, X. Zhou, and J. Yuan, "An intelligent fault diagnosis system for newly assembled transmission," *Expert Systems with Applications*, vol. 41, no. 9, pp. 4060–4072, 2014, ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2013.12.045. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417414000049.

[8] L. Leitner, A. Lagrange, and C. Endisch, "End-of-line fault detection for combustion engines using one-class classification," in *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, IEEE, 2016, pp. 207–213, ISBN: 978-1-5090-2065-2. DOI: 10.1109/AIM.2016.7576768.

[9] S. Gaugel and M. Reichert, "Prectime: A deep learning architecture for precise time series segmentation in industrial manufacturing operations," *Engineering Applications of Artificial Intelligence*, vol. 122, p. 106 078, 2023, ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2023.106078. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197623002622.

[10] A. Sorsa and K. Leiviskä, "Comparison of feature selection methods applied to barkhausen noise data set," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 14 699–14 704, 2011, 18th IFAC World Congress, ISSN: 1474-6670. DOI: https://doi.org/10.3182/20110828-6-IT-1002.01777. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667016459902.

[11] Q.-G. Wang, X. Li, and Q. Qin, "Feature selection for time series modeling," *Journal of Intelligent Learning Systems and Applications*, vol. 05, pp. 152–164, Jan. 2013. DOI: 10.4236/jilsa.2013.53017.

[12] K. Gu, S. Vosoughi, and T. Prioleau, "Feature selection for multivariate time series via network pruning," in *2021 International Conference on Data Mining Workshops (ICDMW)*, 2021, pp. 1017–1024. DOI: 10.1109/ICDMW53433.2021.00132.

[13] J. Li, K. Cheng, S. Wang, *et al.*, "Feature selection," *ACM Computing Surveys*, vol. 50, no. 6, pp. 1–45, 2018, ISSN: 0360-0300. DOI: 10.1145/3136625.

[14] J. C. Ang, A. Mirzal, H. Haron, and H. N. A. Hamed, "Supervised, unsupervised, and semi-supervised feature selection: A review on gene selection," *IEEE/ACM transactions on computational biology and bioinformat-*

ics, vol. 13, no. 5, pp. 971–989, 2016. DOI: 10.1109/TCBB.2015.2478454.

[15] J. Ircio, A. Lojo, U. Mori, and J. A. Lozano, "Mutual information based feature subset selection in multivariate time series classification," *Pattern Recognition*, vol. 108, p. 107 525, 2020, ISSN: 00313203. DOI: 10.1016/j.patcog.2020.107525.

[16] F. Zhu, X. Jia, M. Miller, *et al.*, "Methodology for important sensor screening for fault detection and classification in semiconductor manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, vol. 34, no. 1, pp. 65–73, 2021, ISSN: 0894-6507. DOI: 10.1109/TSM.2020.3037085.

[17] B. Xiao, Y. Li, B. Sun, C. Yang, K. Huang, and H. Zhu, "Decentralized pca modeling based on relevance and redundancy variable selection and its application to large-scale dynamic process monitoring," *Process Safety and Environmental Protection*, vol. 151, pp. 85–100, 2021, ISSN: 09575820. DOI: 10.1016/j.psep.2021.04.043.

[18] S. Agrawal and D. K. Sharma, "Handling high-dimensional data and classification using a hybrid feature selection approach," in *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, IEEE, 2022, pp. 2168–2172, ISBN: 978-1-6654-3789-9. DOI: 10.1109/ICACITE53722.2022.9823878.

[19] H. Bouzgou and C. A. Gueymard, "Minimum redundancy – maximum relevance with extreme learning machines for global solar radiation forecasting: Toward an optimized dimensionality reduction for solar time series," *Solar Energy*, vol. 158, pp. 595–609, 2017, ISSN: 0038092X. DOI: 10.1016/j.solener.2017.10.035.

[20] K. K. Yun, S. W. Yoon, and D. Won, "Interpretable stock price forecasting model using genetic algorithm-machine learning regressions and best feature subset selection," *Expert Systems with Applications*, vol. 213, p. 118 803, 2023, ISSN: 09574174. DOI: 10.1016/j.eswa.2022.118803.

[21] N. El Aboudi and L. Benhlima, "Review on wrapper feature selection approaches," in *2016 International Conference on Engineering & MIS (ICEMIS)*, IEEE, 2016, pp. 1–5, ISBN: 978-1-5090-5579-1. DOI: 10.1109/ICEMIS.2016.7745366.

[22] S. Karasu, A. Altan, S. Bekiros, and W. Ahmad, "A new forecasting model with wrapper-based feature selection approach using multi-objective optimization technique for chaotic crude oil time series," *Energy*, vol. 212, p. 118 750, 2020, ISSN: 03605442. DOI: 10.1016/j.energy.2020.118750.

[23] L. Petrucci, F. Ricci, F. Mariani, and A. Mariani, "From real to virtual sensors, an artificial intelligence approach for the industrial phase of end-of-line quality control of gdi pumps," *Measurement*, vol. 199, p. 111 583, 2022, ISSN: 02632241. DOI: 10.1016/j.measurement.2022.111583.

[24] N. Valceschini, M. Mazzoleni, and F. Previdi, "Inertial load classification of low-cost electro-mechanical systems under dataset shift with fast end of line testing," *Engineering Applications of Artificial Intelligence*, vol. 105, p. 104 446, 2021, ISSN: 09521976. DOI: 10.1016/j.engappai.2021.104446.

[25] V. Hirsch, P. Reimann, and B. Mitschang, "Data-driven fault diagnosis in end-of-line testing of complex products," in *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, IEEE, 2019, pp. 492–503, ISBN: 978-1-7281-4493-1. DOI: 10.1109/DSAA.2019.00064.

[26] V. Hirsch, P. Reimann, O. Kirn, and B. Mitschang, "Analytical approach to support fault diagnosis and quality control in end-of-line testing," *Procedia CIRP*, vol. 72, pp. 1333–1338, 2018, ISSN: 22128271. DOI: 10.1016/j.procir.2018.03.024.

[27] L. Leitner and C. Endisch, "Robust stochastic process models and parameter estimation for industrial end-of-line-testing," in *2018 IEEE International Conference on Industrial Technology (ICIT)*, IEEE, 2018, pp. 1520–1525, ISBN: 978-1-5090-5949-2. DOI: 10.1109/ICIT.2018.8352406.

[28] S. Gaugel and M. Reichert, "Industrial transfer learning for multivariate time series segmentation: A case study on hydraulic pump testing cycles," *Sensors*, vol. 23, no. 7, 2023, ISSN: 1424-8220. DOI: 10.3390/s23073636. [Online]. Available: https://www.mdpi.com/1424-8220/23/7/3636.