Predicting Machine Errors based on Adaptive Sensor Data Drifts in a Real World Industrial Setup

1st Sebastian Soller
Chief Data Officer
Almanara Research GmbH
Ruhstorf an der Rott, Germany
sebastian.soller@almanara-research.de

2nd Gerold Hölzl
Chair of Embedded Systems
University of Passau
Passau, Germany
gerold.hoelzl@uni-passau.de

3rd Matthias Kranz
Chair of Embedded Systems
University of Passau
Passau, Germany
matthias.kranz@uni-passau.de

***Abstract*—We present a dynamic error prediction system for industrial production machines. We implemented a flexible data collection tool to create error warnings for a production line, which aims to improve the already existing static alarm models. For industrial machines, there are threshold-based alarm models set by prior experiences and observations of the operator. For machines without standardized interfaces and communication protocols, which are not Industry 4.0 compatible, it represents a challenge to implement and add a dynamic and opportunistic system behavior. Machines need to learn from past errors autonomously and adapt the production properties dynamically. We implemented a framework that makes production machines conform to the Internet of Things (IoT) concepts, by making previously non-IoT enabled resources available to get new insights into the production processes.**
The system component recognition and the database setup is done fully automatically by our developed system.
We designed and applied a feature-based data drift model in a real-world industrial setting to determine data deviation between normal and erroneous work-pieces in real-time to predict upcoming erroneous behavior. The drift analysis flagged and predicted work-pieces as erroneous several minutes before the pre-defined machine alarms would have been raised. The resulting flagged sensors and values can be compared to the system determined errors to get new insights into the abnormal machine behavior. For the reduction of downtime, the most valuable immediate result of the system is the ability to notify the operator earlier and reduce overall downtime.

***Index Terms*—IoT, Industry 4.0, Middleware Platforms, Self Configuring System, Maintenance Prediction, Unsupervised Analytics**

I. INTRODUCTION

In the modern industrial landscape, there is a vast array of information that can be gathered from the production processes, which is never stored or even reviewed as of today. In industrial settings before the fourth industrial revolution that is happening right now logic controllers control machines. This represents a big step for the industry. The programmable logic controller (PLC) was introduced 1969 and has since become the standard way to control machines in a production line [1]. Those PLCs control the machines and read out sensor values to detect errors in the production process. Errors are generally detected by using pre-defined thresholds, which can be set by a manufacturer or based on the machine operators experience. Through the utilization of IoT concepts and implementations, these static systems can be improve and adapted to become more intelligent to increase efficiency and stay competitive.

The systems we observe need attention at unknown times due to mechanical, electrical or external issues, which need to be resolved manually by a technician. For those errors, the system has to stop immediately and does not operate until the error is fixed on location after an inspection of the reason behind the failure. The summarized downtime consists of the time until the technician is informed of the failure, the walking time to reach the machine and the time on the machine to analyze the error and get it back to working. These kind of procedures can lead to hundreds of hours of downtime over a year and multiple machines.

This study aims to predict the system failure for the technician to be already at the machine site in times of failure or even prevent the events by contradicting the issue early. Time from a failure to the system working properly is reduced by removing the information and walking time of the technician. Currently, the system has no way of tracking the manufactured parts and the results of their quality control at the end of the production process. The possible ground truth that is provided by the system is time since the the last maintenance, which work-pieces were produced and the time of fatal errors, which lead to a machine stop beyond 10 minutes. For the prediction of the maintenance interval, we assume three phases of the machine state. The states are (i) working, (ii) not working and (iii) the phase in between where the upcoming error can already be diagnosed by analyzing the data values and their change.

We conducted a pilot study to gather real-time data of a production machine, using a dynamic iteration to detect all available variables autonomously. The data is analyzed without previous knowledge of the machines variables and what it can provide. We propose a system, that gathers data dynamically and an opportunistic [2] approach to use different techniques of data analysis. The system can generate visualizations for production properties. An outlier detection is used to predict incoming failures of the machine and boost the accuracy and reliability of the alarm system. We a usedm to apply a data drift analysis to detect errors on a data-driven approach. The approach generalizes on to multiple production machines with different purposes. The architecture relies on basic signal

978-1-7281-4657-7/20/$31.00 ©2020 IEEE

processing techniques for a stable and reliable setup. More advanced techniques, that might be more tailored to the specific application cases of the system can easily be added as we dynamically analyse the feature drift in our model. The system fulfills the demands of a low cost, scale-able, self-configuring architecture [3].

II. RELATED WORK

In an industrial environment, information always needs to be transmitted reliable, timely and uniform [4]. For this goal, we need to use a common transmission standard. The network level needs to allow high speed, high bandwidth and real time transmission. This layer needs to be assisted by high-level communication protocols. The applied models need to be backward compatible to include already existing systems, which are in live production as of today. We need to find an intermediate level between the long machine live-cycles inside the industry and the need to evolve the current systems to the next level. New technologies such as IoT are pushing the industry even more towards the smart factory [5]. These improved factories offer mass customization, flexibility for the production process and can optimize management decision making through a higher visibility. These improvements also generate a large amount of data, which we need to handle properly. The data needs to be collected and correctly used to support a proactive maintenance to spare costs and give new insights into the production management. In addition to the important aspects of the technological challenges and standardization challenges, we also need to account for the aspects of security and privacy protection. The standards for cyber security for IoT systems are still vague [6] and further research work needs to be done to guarantee a secure and responsible handling of the data collected. Methods to verify and control those systems, which reach complexities beyond the human ability to monitor system decisions, are necessary. These smart systems and factories need to be robust and resilient for disruptions.

For the predictive maintenance, methods were presented to observe the machine parameters and the product quality at the end [7]. A study about the technical issues, which the predictive maintenance methods needs to address and the importance of these issues for people from the industry, was conducted. The most valuable aspects, rated with the highest priority, were the ability to monitor the process parameters continuously and if those range in between pre-defined thresholds. The main goal is the ability to create a model to predict necessary maintenance (-intervals) and which parts of the equipment need to be exchanged, which would make production decisions easier. Even if the machine parameters for the predictive maintenance did not show a deviation the system can track the work-piece outcome and if there are more erroneous parts than usual. Sensor and quality control data are combined to achieve a sustainable and intelligent system. The model uses machine variables that are selected manually and stores only the selected data, instead of the whole data of the machine, as it is done for the study we perform.

Systems using log data and applying it onto data streams were done before. A log data system to collect data from the running equipment which consists of timestamps, structured text messages describing the event and the event groups, where similar events are grouped together was proposed in previous work [8]. These events are defined by the developer and consist of the values for variables and system states he deemed necessary to report. An experienced worker can use the daily logs to identify critical conditions by manually browsing through the thousands of log messages. We attempt to support these experienced technicians with their task by supplying additional information.

With an evaluation of the existing maintenance records [9] similarly concludes, that it is possible that these maintenances often are not focused on the most critical components. The components are evaluated for their reliability and which components slow the whole process by a low reliability. Their selection for sensors is done by locating and selecting the sensors for the most important components of the system. For these sensor values, they apply previous knowledge about the influence of aspects such as vibration. For every value which is used as a feature there needs to be a range which is considered as normal and a spectrum outside the normal range at which the machine should not be operated dependent on the importance of the component related to the sensor. If we aim to use the methods shown in [9] we first need to extract the important variables for each error from a pool of all variables.

For a pilot study, we accessed an industrial machine for the data collection. The open platform communications unified architecture (OPC UA) standard is the next step of the industrial communication with PLC machines after the previous OPC standard. OPC UA is not limited to a specific platform, such as OPC and uses a standardized service oriented architecture [10]. The main advantage compared to the OPC classic standard is the communication protocol, which is not bound to windows machines and the DCOM standard.

For the purpose of the study we gather data of a single machine from a production line. For the design of the system aspects of a mixed line production, meaning the sequencing of products for the mixed assembly lines is considered [11]. The evaluations include the correct sequence, the number of model types produced on the single line, the deviation of demands each product has for the machine and the models time deviation. Each model, which goes through the process, is dependent on every machine in the product line. For the pilot study, we will also have to handle a single machine inside a production line and deal with unrelated machine stops.

For the analysis pure data based methodologies will be used [12]. Only the machines measurements are analyzed by the system we design. We only use alarm data after the process to verify the results. Data-based methodologies include multivariate statistical analysis, iterative learning and model free adaptive approaches [12]. For the multivariate statistical analysis the principal component analysis (PCA) can be used to reduce the high dimension of data. The variability of the data is observed using the PCA and different groups of data

can be visualized to be compared. The statistical analysis is limited by more complicated non-linear data, which need to be considered by the data analysis of this work. A more complex method is the iterative learning approach. Repeated similar tasks are observed and projected as a dynamic process. For each individual task the information of previous appearances, need to be saved for the iterative learning to work. This requires a well designed storage concept and calculation performance, since big streams of data are compared to each other and calculations have to be made in a live setting [12].

For our proposed system, we need to implement a feature selection for a clear visualization of results. We cannot perform the feature selection by testing different settings and comparing them to a ground truth to evaluate the results, since there is no ground-truth provided for each work-piece. There are methods for unsupervised feature selection. A multi cluster feature selection can be used for the feature selection [13]. This method uses spectral clustering technique to sort the data. For every cluster, a single sparse vector is calculated. The sorting algorithm sorts the features of which an amount should be selected by the maximum entry into the vectors. This method can be useful for this kind of work but needs to be changed since the number of clusters is unknown, or can be defined as the 2 categories (i) running and (ii) stopped. We aim to implement a system, that can differ between normal and abnormal behavior. A system can learn from its normal behavior and then monitor the data differentiation [14]. We aim to build a robust learning system that does not include unknown outliers during the learning process. An important aspect and necessary step to proof any kind of analysis done is the ability to detect statistical meaningfulness. Statistical meaningfulness is a quality criterion for time trends [15]. We aim to calculate important statistics to determine the meaningfulness. The coefficient of determination, which can reach from zero to 100 percent and measures the quality of the fit for a linear regression model, which is fitted to the data. As a second variable, they calculate the p value, which is used in most statistical significance tests. To get the p-value they calculate the two-tailed Students t statistic, which can be related to a p-value if the degrees of freedom are known. They determined 65 percent as a minimum value for the coefficient of determination to be relevant for further significance consideration by the p-value. This percentage was gained by a threshold, which is suggested to be crossed by the predictive power of the linear regression.

III. SYSTEM STATE AND GOAL

We propose the a self-configuring system which automates most used processes with less human steps necessary 1.

We aim to extend the already working manufacturing system of the observed industrial plant. As of now, the system we aim to observe for the pilot study is capable of sending alarms to an operator when machine errors appear. We are not aware of the amount and format of variables, which we can read from the PLC. Our goal is to implement a zero-config, plug and play system. Independent of the machine that the setup gets connected to, the data transmission needs to start automatically. No installation effort is put to the operators. Therefore, we aim to use the announcement properties of the OPC UA server on the production machines. The address space is traversed and every variable which includes readable sensors or programs information values will be saved in a newly instantiated database table at system start time. For each variable, its name, data path inside the machine and datatype are saved and can be linked to the data for later investigation. Using this big amount of data, we aim to predict an upcoming error more accurate and earlier as the current system does by observing single variable thresholds. We will use all sensor readings to predict any alarm. Therefore, we make the following assumptions we aim to validate during the pilot study:

- An upcoming system alarm of the currently used threshold system can be predicted earlier using a data driven approach.
- The data driven approach can give more detailed information about the errors occurring in the system.

In Figure 2 we visualize what we expect our system to achieve. Use all information supported by the machine for all possible system errors to evaluate and boost preexisting systems. The figure shows exemplary the current systems usage of a threshold system by observing specific variables for specific alarm. The system we implement observes all possible variables on their influence on to specific alarms.

IV. METHOD FOR AN ERROR PREDICTION IN AN UNKNOWN INDUSTRIAL SYSTEM

To predict an upcoming alarm we use all readable variables to create features for the prediction, since there is no previous knowledge about variable importance or purpose. We calculate a percentage drift between each work-piece and its successors. We combine the drift for all features into a general system deviation variable. Using the system deviation variable, we compare each work-piece entering the machine to its predecessors and evaluate the magnitude.

A. Feature selection

We defined features from the variables to compare the work-pieces to each other. To achieve a high generalization we used the most common statistical features of the time and frequency domain [16]. The usage of a window function for the feature construction was evaluated. We decided to split the data by the work-piece information for the comparability over the time. Every work-piece of the same type should show similar feature ranges. We create a set of features for every work-piece passing through the machine. The numerical values provided by the system can be used directly to calculate features. For non-numerical variables, we used a hash function to make the calculation possible.

B. Feature drift

We observe a concept drift [17] to raise possible errors. This drift is observable for unexpected data changes. Using

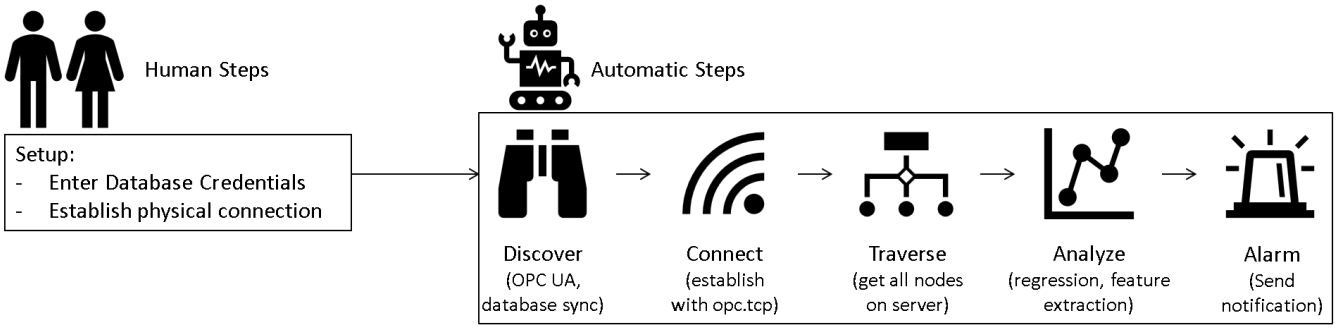

Fig. 1. The system architecture consists of the human steps and the automatic steps. The human steps that are necessary are to enter the database credentials to allow the controller to connect and write to the database chosen. The controller is plugged into the network and powered on (V-A). After the boot the network is scanned for OPC UA servers and the database is checked for already existing connections to any of the found servers by another controller. A connection is established with an unobserved server and the address space gets traversed for readable nodes (V-B). Numerical nodes can be used for the construction of features and analysis such as regression, which can be provided graphically to machine controller if any nodes show positive or negative trends. After data (V-C) is gathered for a short time period the system determines the machines normal behaviour and sends an alarm if multiple values start to show unusual behaviour (VI-B).

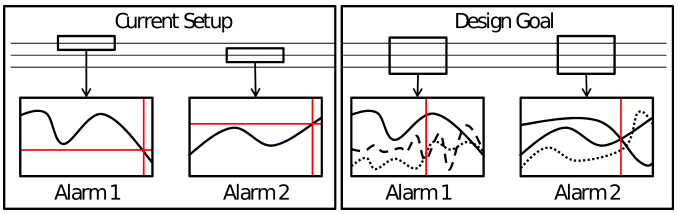

Fig. 2. On the left the current method of using thresholds for different alarms is shown. For a temperature alarm, the temperature is observed for a too high value. On the right side, all variables available are used for a classification. With this method, we aim to predict mistakes earlier and get more insight into the error itself. A temperature error can for example also be caused by measurement failures caused by different aspects of the system, such as the electrical current which can lead to faulty measurements.

the features, we calculate a percentage drift between each value and its successors. The drift for all features is combined into a general system deviation variable. When this deviation system variable peaks, an alarm is sent to monitor the system for an incoming failure or faulty outputs. We calculate each features drift percentage using a window function over the previous work-pieces. A window function with the size n is used to calculate the percentage drift of a feature for the value at position i. The window defines the amount of data point, which we need to calculate the drift variable. The window size should be smaller than the supposed concept drift [18]. We calculate the drift of the data for every feature and work-piece $drift_{ij}$:

$$\frac{value[i] - average(value[i-1] : value[i-1-n])}{average(value[i-1] : value[i-1-n])} \quad (1)$$

The window size for this evaluation is exemplary selected as 10. The window size must be chosen according to two aspects which can be seen in Figure 3. A too short window has two downsides. In Figure 3, using a smaller window, the data points 0 to 15 are weighted stronger and get flattened with a bigger window. For a rising or falling edge, a small window will show the drift to a lesser extent than a bigger window. For a bigger window size, the downsides are the necessity of the window size variables before the method can be used. An extending window for the window size from the beginning can be used, but this results in the downsides of a short window for the beginning values. A bigger window does show a drift after a rising or falling edge for values even when the values remain static afterwards as shown for the data points between 35 and 41. For the method to analyze the drift, peaks in the drift curve are important to recognize relevant changes.

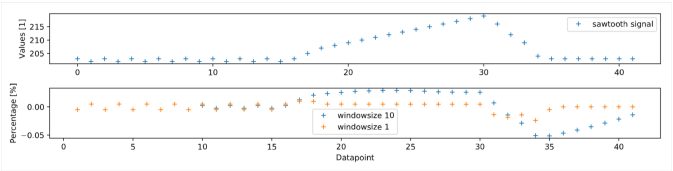

Fig. 3. An evaluation of the window size. A saw-tooth function with some added noise is used to visualize the difference of the window sizes. A window of length 10 and 1 were used to visualize the influence on the signal depending on the window size. For the small window size, long time trends are not observable, as can be seen from sample points 30 to 35, where we cannot detect the falling edge as a drift. Only jumps between two successive data-points can be recognized.

The calculation as presented in (1) is used on all features and results in the matrix $Drift_{ij}$:

$$\begin{matrix} drift_{11} & drift_{12} & ... & drift_{1n} & drift_{1n} \\ ... & ... & ... & ... & ... \\ drift_{n1} & drift_{n2} & ... & drift_{nn} & drift_{nn} \end{matrix} \quad (2)$$

We can work with the drift matrix by selecting the rows or columns separately or iterating for calculations. For every column of the $Drift_{ij}$ matrix we calculate the outliers by using a threshold for the maximum tolerable deviation. We get the threshold by calculating the n-th distance of each work-piece and features measured in standard deviations [19]. We use three times the standard deviation distance from the mean to detect only strong outlier. Features inside a dataset can show drifts due to sensor measurement errors or a high variance

between the work-pieces. For each work-piece, we can sum the amount of features exceeding the defined maximum threshold to get the $work - piece_i.deviation$:

$$1\ if\ drift_{i1} > threshold + .. + 1\ if\ drift_{ij} > threshold \tag{3}$$

For all work-pieces, the average and standard deviation are calculated. If a new work-piece is produced and the deviation exceeds the average plus 3 times the number of deviations an alarm can be sent. The work-piece represents an outlier to the previous production process and we can inspect the system for the underlying reasons.

V. REAL WORLD PILOT STUDY FOR DATA COLLECTION AND ANALYSIS

For the proposed system, we created an implementation for testing and verifying the approach and its assumptions. We evaluated and selected hardware for the set-up according to the needs. We created a system capable of a plug and play function at the machine site in the factory, which starts to gather data without further configuration. The system runs machine independent as long as an OPC UA server is installed to access the data. We designed the system to run maintenance free during the data collection intervals.

A. Hardware Implementation

We tested the setup on a live production line at a production facility. We decided to connect the pilot setup to a production machine in a running production process. The machine already uses alarms, and activates an emergency light when certain variable thresholds are breached. We can use these alarms to evaluate our automatically created alarms. The machine observed is placed inside a production line with preceding processes. A conveyor belt transports the material to the machine. The production program used is running on a PLC. The data was read by using the OPC UA protocol. For the purpose of the study, all accessible variables provided by the machine were gathered. No previous selection was conducted and the available variables were automatically traversed to get all readable data. Using this approach, we can extend the setup to multiple machines and production lines, since the process is configuration free.

For the implementation at the machine site we used an industrial Raspberry Pi 3, the NetPi by Hilscher [20]. A necessary expansion for this study, which the NetPi provides, are additional industrial Ethernet ports on top of one standard Ethernet port. The industrial Ethernet ports can be used as a PROFINET IO-Device, Ethernet/IP adapter or EtherCAT slave. For the hardware installation, we placed the NetPi inside the switch cabinet of the machine. The data is stored locally and the system could be configured to write the collected data into a database.

B. Communication model

The communication between devices takes a key role for our design. For extendibility, a clear communication standard needs to be set. The transmission between machine and controller needs to be exactly defined. Each part of the system has to be able to be globally identified. Especially due to the unknown format and amount of data, no data with unsure origin is allowed into the system. Any production machine in a production line needs to be uniquely identified. The machine needs an IP address and unique MAC ID to get connected inside the network. The MAC ID is used by the whole system to identify the machine inside the network, since IP addresses are able to change. Each machine has an unknown number of variables and is connected to a single controller (the NetPi). To identify each variable inside the machine it needs a unique ID. For the system, the combination of variable ID and the machines MAC ID clearly identifies every variable globally. The variable needs a data type for the correct storage inside the database and temporary on the controller. We installed the database on a central server. For each possible client for the server, rights for accessing the database and necessary tables have to be set. The controller accesses the database by its IP address. For each machine a table inside the database is generated opportunistically, which inherits all values of the machine. This design is used for a set up with a machine using numeric, boolean or string as a data type. For the capabilities of storing images and videos, we would need to extend the system. For the practicality and possible size of these functions, run-time tests are necessary.

We implemented the system, which is currently capable of reading data from an industry machine, saving the data to a database and visualizing the results of the data analysis. Every step of this process is done automatically with the only necessary input of the database IP. The controller is able to be plugged into any network with a running industrial machine with installed OPC UA and known database server. This allows us to extend the system inside the factory. We only need to clone the controller and put it inside the network and it will connect to an unobserved OPC UA server if one exists inside the network. The selection of data variables of the machine will be done dynamically by the system. The base sequence of the events of the system is according to Figure 4. The communication sequence is a linear process. The machine side should have the ability to communicate each possible variable to the controller. The controller requests all possible variables. If no complete list is available by the machine the controller needs to browse through the data structure of the machine to get the information needed. The controller should check the database for an already existing table or file, which is identical to the naming scheme of the current machine. If a table exists, the gathered values are written into the table instead of creating a new one.

For this approach, security concerns have to be considered when the system is extended in scale, considering network attacks or the possibility of the database being accessed by manipulated data values. For the pilot study, we used a closed network, which only inherited the industrial machines and the database with no possibility of an outside connection. The machine networks are also separated by using Docker

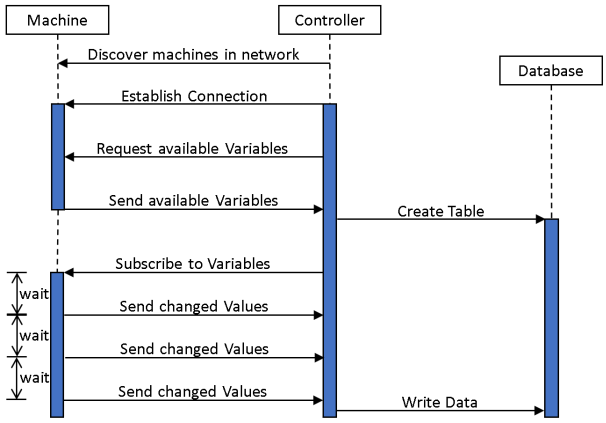

Fig. 4. The the connection between the NetPi controller and the OPC UA server is established dynamically. The connection to a not already monitored machine in the network is established. A connected controller sets a connection flag inside the machines database. A new introduced controller will not connect with already observed machines. The controller establishes a connection to a new OPC UA server and browses the server to get all available data variables. After all variables are received, a table is generated on the database server. The Machine server sends the data for subscribed variables periodically to the controller, when a variable change took place during the interval. The data are synchronized to the database when the local storage size reaches a pre-defined value.

containers [21] for different tasks. One container runs the functionalities towards the industrial machine and writes the data locally. The second container collects the locally stored data and sends it periodically to the database. This way we can control that the machine is not directly connected to any network.

C. Gathered Data

The resulting dataset consists of a table with a column for each variable sampled at 10 Hz. The data types vary for each column. The dataset includes values for sensor variables, such as temperature or vibration and information about the running PLC program, such as program name or product type name and system alarms. The data amounted to around 1.5 GB daily, which summed up to 90 GB over two month. Reading from an informative line of strings it can be determined whether or not the machine is stopped and thereby, when a single parts production starts or ends. This makes it possible to differentiate between each work-piece form the same product group for a separate analysis. We split the data into frames for each type and an additional type, stopped, where the data between the work-pieces are saved. For easier tracking, we added the number of the work-piece for the prototype study. We enumerated all work-pieces as well as the stops, for easy sorting of which work-piece was the following and succeeding, on top of the already existing timestamp. There is a stop between two work-pieces for every instance of the pilot study. We enumerated the stops separately for the work-piece types. Work-piece number X is followed by stop X inside the data. For the comparison of similarity between the correct signal and a new incoming signal different metrics, such as the root mean squared error (RMSE) or median absolute error (MAE) can be used [22]. From a optimal curve, the statistical features for every variable and product type can be calculated 5. We can uniquely identify each product type by its data value curve using a classification. As evaluation, we used the data values for a classification of the product types. A J48 decision tree was used for classification with a tenfold cross validation. The result was the correct classification of 21020 instances out of 21413, which is an accuracy of 98.2 percent. Inside the data set were 17 different product types as classes.

For every product type and variable measured from the machine an optimal data curve can be derived. For each product type, we selected all work-pieces that do not deviate in length from each other. By analyzing all product types only an average of 1.1 percent deviate in length from all other work-pieces of the same product type. Any other work-pieces are similar in length with a maximum deviation of one second. The average curve is calculated by the average of each work-piece for each value of the time series. The average is surrounded by the standard deviation as can be seen in Figure 5.

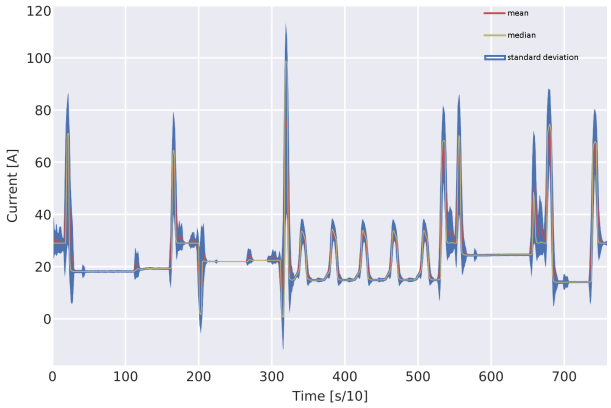

Fig. 5. The average curve for a product type and a single feature. The orange line marks the average position of the signal with the standard deviation for each position around it in the dark blue color. We generated the curve using 254 instances of the work-piece.

D. Regression

We use a linear regression of grade one to detect trends inside the data. Every feature is observed separately. The results for the regression and its r-squared value differ between analyzing the whole data set or segments of data and product types. We applied the setup to each variable and a subset of variables using the PCA. With the regression analysis, we aim to find trends.

We use this linear predictor to analyze the statistical features constructed. These predictions can be compared to manually set thresholds or thresholds defined by system crashes. An optimal system will not show any deviation. As a validation method, it is not possible to test the result against a ground truth. There needs to be a measure for change of a raw data column or feature over time, which shows deviation from previously measured values. In the first step, we performed a regression on the two-month data set. The data were right through optical inspection chaotic and the regression does have

a low coefficient of determination (r-squared) if used on itself. For the work-piece dataset, the r-squared has a value of 0.08. We interpreted that the model does not project the variability around the mean of the data well.

To analyze the data further and explain several peaks, we add the pause timing. Every stop, which exceeded one hour, we used to split the data into segments. As a result, the data is split into three segments from one continuous data stream of the recording period. For the regression of the three segments, we get r-squared values of 0.1 for segment one, 0.31 for segment two after the long break and 0.27 for segment three. This already shows an improvement to the setup without any segmentation.

For multiple variables, the values for various statistical values are within different ranges for different product types. This leads multiple variables to be of no use for any kind of regression. Product types alternate between each other in a mixed production, which is normal for this production line. By separating the product types, we can observe and analyze trends for each type. The average coefficient of determination for each segment, when the product types are used separately are 0.39 for segment one, 0.50 for segment two and 0.63 for segment three. The regression results for all segments are better at representing the data if we split for the segment and product type. This shows that the machine stops strongly influence the production machine. The machine, in these times, can cool down and many variables are reset to the status identical to directly after maintenance. For long time trend analysis, this poses a problem because it can mask an actual significant trend. It would be necessary to separate for the product type. Due to the short windows of a single type compared to the long time analysis, this makes the regression approach difficult. To observe the trend for each product type with a high enough r-squared value the slope needs to be tracked and when the overall slope for all product types shows the same tendency it can be concluded, that the machine does deteriorate over a long time.

VI. ANALYSING THE FEATURE DRIFT FOR ERROR PREDICTION

For the error prediction, we do not use the information we can achieve by optical inspection or findings during the manual visualization process. This allows us to apply the model on further machines with less manual work. We can use any statistical features from the time and frequency domain for the drift calculation. For the example presented, we kept the amount of projected features small for an easier visualization of the results. We used the mean and the variance of the data stream as the two features of the test setup for general usage with all numerical data. An exemplary data exempt can be seen in Figure 6 for one of the features.

A. Variable Drift

For the windowed data we get the values of percentage deviation for each sample for a feature in Figure 7. For all features, these values were calculated. For most features, a small drift for the most work-piece was observable. Each feature drift showed peaks at different times. To compare the values to each other, normalization was used. This had to be done due to the different value ranges. In example, one feature we observed with values ranging from 2.0 to 3.0 and dropping to 0.8 for a work-piece. Another feature ranged from -346 to -348 and dropped for the error to -365. The drift for feature one would be 0.68 and for feature two 0.05. For a normalized calculation the drift for example two can be observed to be much bigger, since the actual range of the values is 19 and the normalized values are 0 to 0.11 for the normal values and 1 for the erroneous value drifts by the factor 19 and will be recognizable compared to the 0.05. The method was tested for the data with and without normalization and the results achieved are comparable. To keep the approach as general as possible normalization should be used for a general implementation.

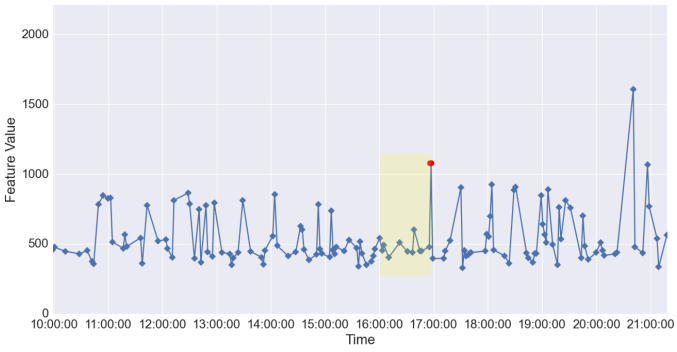

Fig. 6. A statistical feature calculated from the data. The feature is calculated using the average per work-piece for the variable of a sensor measurement of the machine. Every single value, one exemplary marked red is compared to its preceding n values and its percentage of change is measured.

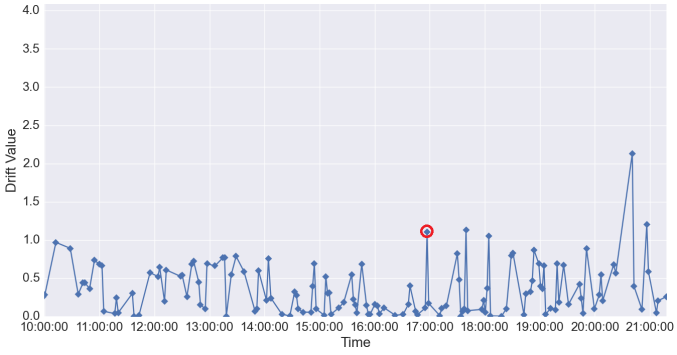

Fig. 7. Each sample in the graph displays the difference of the sample on this position to the 10 succeeding samples. The marked sample is the result for the marked sample of Figure 6. The figure shows a drift for more then 1 which means a change of the value of more than 100 percent for multiple work-pieces of the feature. This can be attributed to the overall variance of the features and this feature in particular. If the single feature would be used to set up an alarm based on this threshold, multiple false errors would have been sent.

B. Drift Magnitude

We analyze the deviation for all features and variables. As a threshold for the identification as an outlier, we selected a drift of more than 50 percent. There are features that are changing often due to their regular behavior and are by themselves not meaningful for any form of classification. These single outliers result in the wrong detection of errors when used as a single measure to detect an error. For each work-piece, we conducted a count of outliers to evaluate the amount of deteriorating features for each point in time. The count of outliers for all work-pieces before the alarm is projected in Figure 8. To evaluate a clear change in the data, an unusual high amount of deteriorating features has to be recognized. We calculated the mean of the number of strong value changes and the median absolute deviation from the counted features. We use the factor of plus or minus three for the median absolute deviation to recognize any data as an outlier [23]. As a result, we identify several work-pieces before the alarm, which are behaving unusual, as critical. The alarm could have been predicted from this point in time and a warning can be sent based on the algorithm. In the current machine, a system value triggered a threshold based alarm, which was set based on pre-defined limitations of the machine. The alarm informed on a later point in time as the actual start of the deviation. The following work-pieces after the alarming drift appear to have a low number of drifting features since the faulty measurements are compared to previous erroneous behavior. A technician can investigate

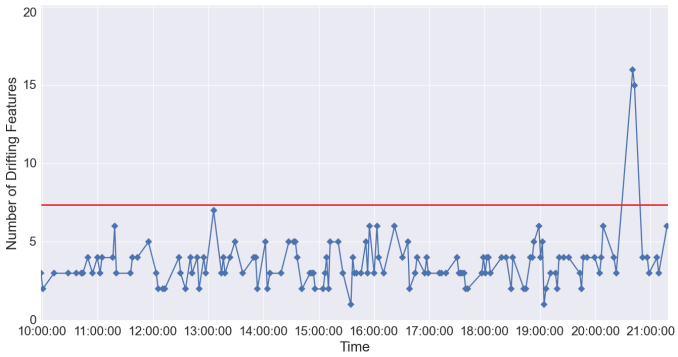

Fig. 8. For every feature we calculated, when the feature has a drift higher than half of the variable range (50 percent drift or more). The sum of features that exceed the threshold are counted and projected in this figure. On average over the work-pieces, we observe for this graphic 3.9 features show a drift of over 50 percent. The median absolute deviation for the data is 1.2. The red line shows the limit for the maximum deviation that is allowed before an alarm is sent. The earlier outliers for single features as in the feature of Figure 7 are not mistaken for errors. A high amount of features starts deviating half an hour before the machine alarm was sent. For the exemplary alarm, a warning could have been sent out earlier for a short manual checkup instead of a long repair interval.

the error afterwards to get information in detail. The name of the features that were responsible for the earlier recognized drift error and its underlying system variable are available. The system saves all information the OPC UA server supports. The variable number can be matched to the variable name. All variables that caused the alarm to trigger are observed and conclusions can be drawn.

TABLE I
AN EXEMPLARY DATA SET FOR THE CLASSIFICATION. USING THE OBTAINED INFORMATION, WE ASSIGN CLASS LABELS TO THE DATA.

Feat. 1	Feat. n	Drift Magn.	Time to failure	Incoming Error
0.1	0.1	5	Good	Good
0.1	0.3	6	Good	Good
0.1	0.1	5	-5	Incoming
0.1	0.2	16	-4	Incoming
0.1	0.1	15	-3	Incoming
0.2	0.2	5	-2	Incoming
0.8	0.1	5	-1	Incoming
0.8	0.1	6	Alarm	Bad

C. Advantages and Categorization

We observe the alarms our system sent and compare the information to the threshold-based alarm. We could detect system sent alarms earlier. The time saved for the detected alarms was on average 15 minutes, which could be enough to be on location at the moment of occurrence. Especially for detected electrical or sensor measurement errors, this method would have saved production time. These kinds of errors take an experienced worker seconds to recognize as a fault and only need to be checked as OK on a terminal for the production to resume. Additionally, these kinds of errors could be examined through the system variables. The temperature sensor surpassed the threshold reached for an exemplary temperature error. All previous variables, which caused the flag by our system where connected to the electrical current and other sensors changing in behavior. We can conclude that the temperature did not cause the actual error, but different erroneous behavior of the machine. We can use the time between the first detected work-piece by the drift method and the work-piece, which the threshold alarm system flagged to create labels for each work-piece. We can apply the labels different to the work-pieces depending on the method we want to use for prediction of the future. We can label the data as can be seen in table I. The first attempt is to analyze between good values and starting from the first deviation in number of work-pieces to the error. We call this method time to failure. For the second method, we mark previous values as good and the work-pieces between the data deviation start and alarm are marked as incoming error. The assigned labels can be used to implement a probabilistic model to train a prediction method. The certainty of these predictions can be used to support the decisions by either the technical experts or the management on whether or not to act on a prediction [24].

With these assigned labels, we can create profiles for each error and analyze if an identical machine behavior leads to identical errors. For the most common error, which we detected 14 times during the test period, we could detect a characteristic change inside the data 11 times before the alarm. For this specific behavior, we can use a classifier on the data stream to detect these characteristic changes over multiple work-pieces. A further data collection and more instances of different sparse errors can be used to train a classification system. This system can send warnings earlier using the data

features and the drift magnitude if the error is recognizable from the data stream.

VII. CONCLUSION AND FUTURE WORK

For the classification, we created the inevitable and needed labels for the data. Using our system, we assign these labels on the fly during the data recording. They are dependent on alarms and the analysis of the data drift. To analyze the data drift is a novel approach to apply labels to unlabeled data. We can train a classification using these labels and depending on the classifier, predict the probability for a system failure. The system is used as of today, to send warnings when a drift magnitude error. An alarm needs to be checked by a technician after the reception. We can input importance of the alarm for a probability model to react and send an alarm depending on the certainty for the error. At the end of the pilot study, the amount of machine errors was low, because of the low error rates machines have in optimized industrial settings. From the recorded errors, we can see that data values deviate before the actual alarm. For an actual measure of accuracy for the classifier, this is not enough data, because for every error occurring, there is no test and train set, but only sparse occurrences. For the analysis of the classification, more data can be gathered and the data drift can be used to apply labels to the data. For the errors observed so far, we can conclude that the alarms triggered by the threshold system react later to a deviation compared to the alarms triggered by implication inside the data. The goal we set to inform the technician before the alarm happens is possible. The alarms so far are set due to thresholds for specific sensor values. Any alarm observed in the unimproved system, could have been detected using different values or especially a combination of variables and their features. The implementation of a probabilistic model for the alarm detection can reduce the downtime. A live analysis system, as proposed by this work, will reduce downtime costs and support a better understanding of the machine behavior in the future. Based on our drifting feature approach, it is possible to analyze unlabeled data streams and compare them to the expected behavior. This provides an unsupervised way to identify changes in the machine produced process data and react accordingly. The developed system is a zero-config, opportunistic one, thus it generates the whole processing and analyzing chain just by plugging it in. This is a tremendous improvement in usability and efficiency as no experts are needed that install and maintain the system nor to interpret its data.

REFERENCES

[1] K. Henning, "Recommendations for implementing the strategic initiative industrie 4.0," 2013.

[2] G. Hoelzl, M. Kurz, and A. Ferscha, "Goal oriented recognition of composed activities for reliable and adaptable intelligence systems," *Journal of Ambient Intelligence and Humanized Computing (JAIHC)*, vol. 5, no. 3, pp. 357–368, July 2013.

[3] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approaches," *IEEE Transactions on industrial electronics*, vol. 56, no. 10, pp. 4258–4265, 2009.

[4] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 17–27, 2017.

[5] F. Shrouf, J. Ordieres, and G. Miragliotta, "Smart factories in industry 4.0: A review of the concept and of energy management approached in production based on the internet of things paradigm," in *Industrial Engineering and Engineering Management (IEEM), 2014 IEEE International Conference on*. IEEE, 2014, pp. 697–701.

[6] L. D. Xu, E. L. Xu, and L. Li, "Industry 4.0: state of the art and future trends," *International Journal of Production Research*, vol. 56, no. 8, pp. 2941–2962, 2018.

[7] J. Lindström, H. Larsson, M. Jonsson, and E. Lejon, "Towards intelligent and sustainable production: combining and integrating online predictive maintenance and continuous quality control," *Procedia CIRP*, vol. 63, pp. 443–448, 2017.

[8] R. Sipos, D. Fradkin, F. Moerchen, and Z. Wang, "Log-based predictive maintenance," in *Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2014, pp. 1867–1876.

[9] D. C. Swanson, "A general prognostic tracking algorithm for predictive maintenance," in *Aerospace Conference, 2001, IEEE Proceedings.*, vol. 6. IEEE, 2001, pp. 2971–2977.

[10] S.-H. Leitner and W. Mahnke, "Opc ua–service-oriented architecture for industrial applications," *ABB Corporate Research Center*, 2006.

[11] E. M. Dar-El and R. Cother, "Assembly line sequencing for model mix," *The International Journal of Production Research*, vol. 13, no. 5, pp. 463–477, 1975.

[12] S. Yin, X. Li, H. Gao, and O. Kaynak, "Data-based techniques focused on modern industry: An overview," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 1, pp. 657–667, 2015.

[13] D. Cai, C. Zhang, and X. He, "Unsupervised feature selection for multi-cluster data," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 333–342.

[14] M. Jain, M. Gupta, A. Singh, and V. Chandan, "Beyond control: Enabling smart thermostats for leakage detection," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 1, p. 14, 2019.

[15] A. C. Bryhn and P. H. Dimberg, "An operational definition of a statistically meaningful trend," *PLoS One*, vol. 6, no. 4, p. e19241, 2011.

[16] D. Figo, P. C. Diniz, D. R. Ferreira, and J. M. Cardoso, "Preprocessing techniques for context recognition from accelerometer data," *Personal and Ubiquitous Computing*, vol. 14, no. 7, pp. 645–662, 2010.

[17] I. Žliobaitė, M. Pechenizkiy, and J. Gama, "An overview of concept drift applications," in *Big data analysis: new algorithms for a new society*. Springer, 2016, pp. 91–114.

[18] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 97–106.

[19] D. Cousineau and S. Chartier, "Outliers detection and treatment: a review," *International Journal of Psychological Research*, vol. 3, no. 1, pp. 58–67, 2010.

[20] Hilscher Gesellschaft fuer Systemautomation mbH. (2018) Netpi. [Online]. Available: https://www.netiot.com/netpi/industrial-raspberry-pi-3/

[21] C. Boettiger, "An introduction to docker for reproducible research," *ACM SIGOPS Operating Systems Review*, vol. 49, no. 1, pp. 71–79, 2015.

[22] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate research*, vol. 30, no. 1, pp. 79–82, 2005.

[23] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, vol. 49, no. 4, pp. 764–766, 2013.

[24] L. Uusitalo, A. Lehikoinen, I. Helle, and K. Myrberg, "An overview of methods to evaluate uncertainty of deterministic models in decision support," *Environmental Modelling & Software*, vol. 63, pp. 24–31, 2015.