Flooding Prevention in Distillation and Extraction Columns with Aid of Machine Learning Approaches

Jonas Oeing*, Laura Maria Neuendorf, Lukas Bittorf, Waldemar Krieger, and Norbert Kockmann

DOI: 10.1002/cite.202100051

Flooding of separation columns is a severe limitation in the operation of distillation and liquid-liquid extraction columns. To observe operation conditions, machine learning algorithms are implemented to recognize the flooding behavior of separation columns on laboratory scale. Besides this, the investigated columns already provided the modular automation interface Module Type Package (MTP), which is used for data access of necessary sensor data. Hence, artificial intelligence (AI) tools with deep learning offer high potential for the process industry and allow to capture operating states that are otherwise difficult to detect or model. However, the advanced methods are only hesitantly applied in practice due to complex combination of operational sensing, data analysis, and active control of the equipment. This article provides an overview on how AI-based algorithms can be implemented in existing laboratory plants. Process sensor data as well as image data are used to model the flooding behavior of distillation and extraction columns for stable and robust operational conditions.

Keywords: Clustering, Convolutional neural networks, Flooding, Process monitoring, Time series forecasting

Received: May 03, 2021; *accepted:* September 27, 2021

1 Introduction

With upcoming data science and neural network knowhow, a lot of application fields are emerging in the process industry. In order to use these new methods (e.g., for control algorithms), it is reasonable to gain experiences with small scale apparatus, but comparable processes already in the laboratory. The following brief overview shows that there are good examples for procedures and interoperability when adopting machine learning (ML) solutions. The main focus is set on an easy-to-follow procedure for the integration of ML solutions and combine these solutions with existing concepts of service-oriented architecture as well as the manufacturer independent interface module type package according to VDI/VDE/NAMUR 2658.

ML algorithms are widely employed in process engineering and often hidden behind the terms of artificial intelligence (AI) [1], soft sensing [2], data fusion [3] or digital twin [4]. In the process industry, typically, large amounts of data are available from the physical sensors in a production plant [5]. This data can be collected, combined, and processed by means of ML algorithms (data fusion, soft(ware) sensors) to obtain more meaningful data for the training of data-driven models that capture the real process conditions [2, 3]. The virtual model (digital twin) can be further used to optimize the production process and an implementation in a loop with the physical world ensures the adaptability of the model [6]. Commonly, one of the following three goals is pursued when applying ML techniques in the process industry: (i) online prediction, (ii) process monitoring, (iii) process fault detection [2]. Some of the most popular algorithms to achieve these goals are provided in Tab. 1 [7]. The ML algorithms are classified into supervised or unsu-

Table 1. Examples for popular unsupervised and supervised ML methods.

Unsupervised learning	Supervised learning
Principal component analysis	Partial least squares
K-means clustering	Support vector machine
Kernel density estimation	Decision tree
Self-organizing map	Artificial neural network

Jonas Oeing, Laura Maria Neuendorf, Lukas Bittorf, Waldemar Krieger, Prof. Dr.-Ing. Norbert Kockmann
jonas.oeing@tu-dortmund.de
TU Dortmund University, Department of Biochemical and Chemical Engineering, Laboratory of Equipment Design, Emil-Figge-Straße 68, 44227 Dortmund, Germany.

All authors contributed equally.

pervised depending on training data being labeled or unlabeled, respectively. The working principles behind those algorithms with some example applications are concisely summarized in Ge et al. [7].

Principal component analysis (PCA) and partial least squares (PLS) are particularly popular as they are easy to implement and capable of tackling a variety of problems, e.g., reduce the dimensionality of the data, extract key features and detect outliers. Hence, it is not surprising that these algorithms make up for the majority of applied ML models that are described in literature [7]. In combination with a regression model, reliable online predictions with high-dimensional data become available [2]. Often used regression models include support vector machines (SVMs) [8] and models based on decision trees (e.g., gradient boosting regressor (GBR), ADABoost or random forest) [9–11]. The advantages of these regression tree ensembles include fast training times and the ability to handle large amounts of data, while providing good accuracy due to combination of multiple estimators [9]. This decision is also made to facilitate the integration of adaptive solutions in future, which would require repeated training of new models, e.g., via recursive or ensemble-based methods [12]. However, individual regression trees are usually not competitive with other methods like SVMs or neural networks [13], but thanks to the low computational cost, regression trees can be combined with bagging or boosting techniques to build a group of estimators to improve predictive accuracy and control overfitting [14, 15]. In bagging, each estimator is trained on a subset of data and the output of every estimator is averaged for the final prediction, e.g., random forest [15] or extra trees regression [13]. In boosting, "weak" estimators are trained in succession on a subset of data and combined into a single "strong" estimator. This can be achieved, e.g., by weighting the weak estimators according to their accuracy (AdaBoost) [14] or by fitting the weak estimators using an arbitrary loss function (gradient boosting) [16]. Clustering is another popular method that is used to organize unlabeled data according to their similarity. Combined with PCA it is a common method for process monitoring and process fault detection [7].

The aforementioned algorithms belong to the classical ML techniques and can be quickly implemented as they are readily available in software libraries for, e.g., Matlab, Python or R [17]. The following procedure to employ ML techniques in a digital twin framework was demonstrated by Min et al. and can be roughly applied in most cases [6]:

- data preprocessing
- feature extraction
- model training and validation
- tryout and optimization
- online deployment

Preprocessing describes, e.g., the temporal alignment, denoising or scaling of data [2, 18], feature extraction, the selection or transformation of data by means of a correlation matrix [19], PCA [20] or even operator experience [6]. Some common modeling techniques were already presented in Tab. 1. Other noteworthy approaches include genetic/evolutionary algorithms, fuzzy logic, probability-based techniques (e.g., Gaussian processes, [21]), semi-supervised and reinforcement learning or artificial neural networks. For the latter kind, structures with convolutional layers and long short-term memory units (LSTM) have proven to be effective for image analysis and time series data, respectively. These neural networks capture the spatial or temporal structure of the data and led to the remarkable advances in image classification and speech recognition [22]. As the vast number of choices for a ML model can be overwhelming at first, it is common practice to test different models and compare their performance based on chosen metrics. In case of a regression problem, the root mean squared error or the coefficient of determination (R^2) are often used. If an adaptive online mechanism is desired, the training and forecast speed can be a deciding factor as well.

During the tryout and optimization stage, the trained model is tested in a real-time operating environment. Since the model is trained on historical data, it is important to verify its operational reliability in the latest environment and adapt the model if necessary. Finally, the virtual model is deployed with connection to the real-time data and the process control or monitoring system. The optimal set of control parameters can be found by means of search algorithms like depth-first search, breadth-first search or grid search [6] or by employing model predictive control or other control strategies [2, 19]. For security reasons, it is advisable to implement visible recommendations from the ML model for an operator rather than a direct access to the process control system, especially at an experimental stage [23, 20]. This is related to the veracity problem that is often associated with ML solutions [24]. It can be difficult to generate interpretable suggestions made by purely data-driven models and justify the adaptation of, e.g., a control strategy based on these models. A possible solution lies in the incorporation of first-principle models to form hybrid models that support rational decision-making [20, 25]. Such advances could revolutionize the perception of ML solutions in process engineering, but are still considered as a "long, adventurous, and intellectually exciting journey" [25].

Another opportunity exists in the integration of edge and cloud computing solutions for the facilitated access and treatment of data [23]. With the increasing computational power of microcontrollers, it becomes more and more possible to locally preprocess and analyze sensor data, e.g., for each unit operation and forward the processed information to a higher-level control or data handling structure, which in turn can function more efficiently due to the reduced amount, but higher quality of data [3, 23]. The development of this type of "smart equipment" by using new sensor or ML solutions to determine hard-to-measure variables and offer more process flexibility has attracted considerable attention in academic and industrial research [1, 4, 26].

Some examples include the digitalization of extraction columns by means of novel measurement techniques complemented by modeling and simulation methods to create tools for predictive online monitoring, which is reviewed in Hlawitschka et al. [27]. Other approaches work on the integration of novel sensors and actuators to obtain valuable process information and create more responsive equipment [28]. In order to facilitate the integration of the given examples and other ML solutions into existing processes, new concepts with standardized interfaces and communication protocols are emerging [29]. One promising concept that stands out in these aspects is the module type package (MTP), which is a module-based approach with embedded process knowledge and standardized interfaces according to VDI/VDE/NAMUR 2658 part 1–4 [30–33]. MTP enables a quick and flexible design of processes and integration of modules, so-called process equipment assemblies (PEAs), compare with VDI 2776 [34], into a higher-level control system, which is referred to as process orchestration layer (POL) [35]. Due to the standardized interfaces, the data from every PEA or the entire process is easily accessible for devices communicating within the network of this plant. Hence, for a device directly communicating with the PEA, ML solutions can be quickly implemented. In order to make these data also accessable to software located outside of the field-level, e.g., a cloud application, it is referred to the NAMUR Open Architecture concept (NOA) [62]. A special MTP feature is the service-oriented architecture that provides the possibility to run recipes with the predefined services each PEA offers to the POL [36]. This feature could be used to run the process with many different control variables, study the respective effects of control variables and observe states that are usually undesired. For most processes this kind of data is scarce as it is not the optimal way to operate the process, but it is useful for the training of ML models that are supposed to prevent those states [20]. The recipe feature already existing in modular automation and MTP context could also be further used for the automated conduction of experiment plans that were designed via design of experiments (DoE). Additionally, ML algorithms could be used subsequently to optimize a product or process [37]. The implementation of such ML solutions via a service-like architecture as proposed by Soto et al. [38] is also an interesting concept, which would greatly benefit from accepted standards.

The aim of this work is to build upon these good practices and provide additional guidance for the implementation of ML methods based on another use case, i.e., flooding prevention in a spinning band distillation and an extraction column. For a distillation column, sensor data is evaluated to forecast the pressure drop with supervised learning methods. Subsequently, the operating state is classified based on the forecast combined with a clustering algorithm to form an early flooding warning system. The distillation column is designed according to the MTP (module type package) concept, which provides easy access to all sensor data from OPC UA servers via a Python script, demonstrating the advantages of standardized modular equipment including its interfaces. Flooding in the extraction column is analyzed via computer-vision and a convolutional neural network to design smarter equipment that can identify its own operating state. Finally, an online control strategy based on the ML models is discussed and its usability within the MTP architecture is evaluated.

2 Experimental Setup

The laboratory experiments were performed with two separation columns for distillation and extraction, respectively. Both are equipped with a newly developed, modular automation system following the MTP approach [47].

2.1 Spinning Band Distillation Column

The investigated spinning band distillation column (SBDC) is built as a modular process equipment assembly according to VDI 2776. It consists of a DN25 glass column with a solid rotating internal – the spinning band (Fig. 1, right) manufactured by Normag-Pfaudler, Ilmenau, Germany. Hence there is only a thin gap for the countercurrent liquid-vapor flow between the spinning band and the wetted glass wall of the column. The rotation induces an intensified mass transport between liquid and vapor, resulting in better separation efficiency with higher rotation speed. On the other hand, increasing the band speed, pressure drop is increasing due to higher liquid loading and higher vapor velocities in the column. At a certain point, depending on band speed, liquid loads, and vapor velocity, the liquid is accumulating, which results in flooding. For detailed information about the spinning band column and its behavior see [39, 40, 63, 64] Flooding can be seen visually but also detected by very steep rise of the pressure drop, which is measured via a pressure difference transmitter between bottom and head of the distillation column.

In literature, several phenomena are described leading to flooding behavior in distillation columns. Depending on the column internals – tray or packing – the flooding mechanisms differ, but in the end, it is always an excessive accumulation of liquid at a specific location that negatively affects the desired separation process [42, 43]. In case of the spinning band distillation column, flooding always occurs below the feeding zone due to the highest local liquid load at this point. Since the pressure drop for this flooding mechanism always behaves very similar for this situation, regardless of whether the situation was caused by high rotation band speed, liquid load or evaporation rate, observation and prediction might be easier compared to industrial columns. Besides the already mentioned actuators and sensors, the column is equipped with necessary sensors and actuators to be operated nearly fully automated. The automation is executed on a PLC by WAGO Kontakttechnik GmbH & Co. KG, Minden, Germany, and features the man-

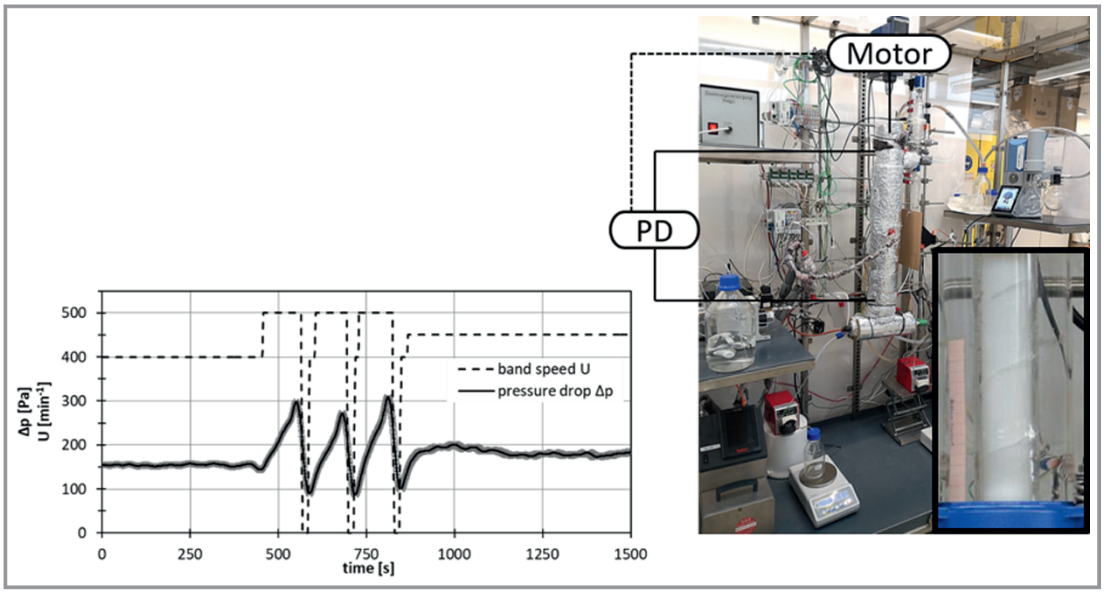

Figure 1. Normal flooding behavior with increasing band speed (left), spinning band column with a view of the uninsulated column with vacuum double jacket and internal (right) [41].

ufacturer independent interface module type package (MTP) standardized in the VDI/VDE/NAMUR 2658 guidelines. Besides this, the distillation column offers a service "distill" with a state machine implemented in the decentralized logic of the modules PLC. Enabled by the MTP, the distillation column can be integrated to the prototypical Process Orchestration Layer (POL) by ABB Research Center, Ladenburg, Germany, where the service can be conducted and the human machine interface (HMI) is automatically visible. Deeper insights of the MTP concept and its architecture can be taken from [35, 44], the architecture combined with the ML implementation is explained in more detail below.

Regarding the flooding phenomena, the distillation column is behaving similarly to conventional distillation columns. The pressure drop is increasing nearly linearly with the gas factor and liquid loading. Beyond certain gas factors and liquid loadings, called loading point, the pressure drop is increasing more steeply than before until the flooding point is reached, which is an undesirable state of the column.[45, 46] In the case of the spinning band distillation column, the pressure drop is also massively influenced by the spinning band speed. This behavior could be already implemented to a control strategy [47], but still, the column is running into the undesired state of flooding. The complex hydrodynamics of the system with a rotating internal and the two-phase flow is very hard to model and predict precisely. Hence, the flooding points need to be examined experimentally. This procedure is very time consuming and still has some uncertainties, which cannot be described completely. Thus, it is of high interest to predict the pressure drop and classify the current operating point with help of a trained ML algorithm with historical data. Like this, it would be possible to already get information about a certain parameter set and a classification even before undesired states like the flooding point is reached in operation.

2.2 Liquid-Liquid Extraction Column

A liquid-liquid extraction column serves the purpose of separation of solvent mixtures with the help of a third solvent immiscible with the carrier solvent [48–50]. In the investigated case, a value component dissolved in a light phase is contacted with the heavy phase, from which it can be separated more easily in a following process step. Butyl acetate is used as the organic phase and deionized water is used as the heavy, aqueous phase. Two operating states within an extraction column can be identified. The regular operating state, characterized by a high separation efficiency through a large mass transfer of the solute from light to heavy phase. On the other hand, the separation efficiency decreases significantly as flooding, the second but undesired operating state occurs, and the volumetric throughput breaks down.

As the laboratory extraction column is optically accessible, a different approach based on computer vision is implemented for flooding detection [51, 52]. Here, image or video data is fed into a deep learning algorithm, i.e., a convolutional neural network (CNN). The network's objective is to distinguish between the desired normal operating mode and flooding of the column.

The investigated DN15 glass extraction column is equipped with a stirrer HS-30D from WiseStir and a pulsation unit HT-120DX from Witeg (Fig. 2, left side) to optimize and control the performance of the extraction. The overflow is used to set the height of the liquid-liquid phase boundary in the column's head.

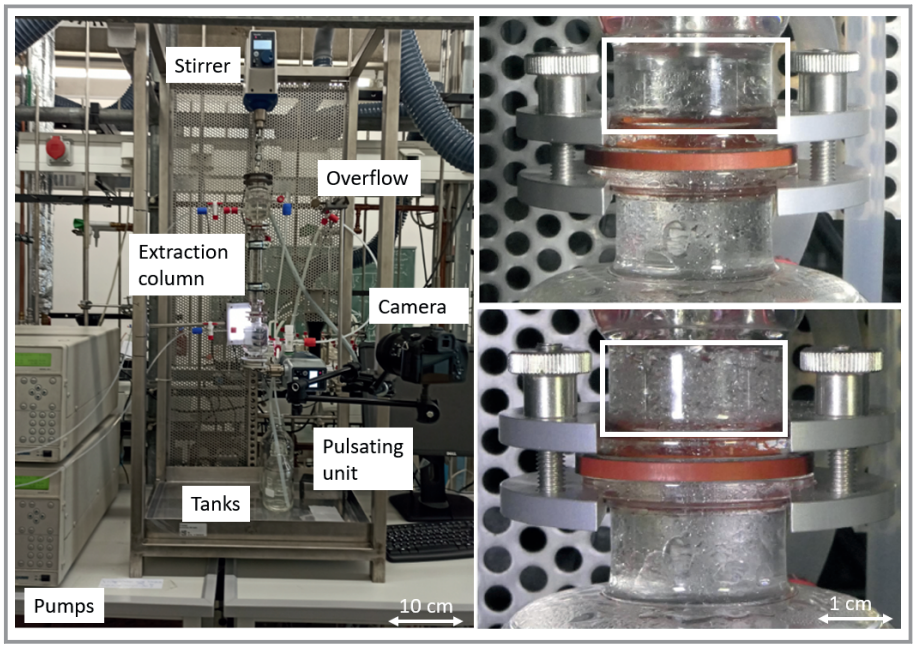

Figure 2. Image of extraction column (left-hand side), example pictures for optimal operating point (upper right-hand side) and flooding (lower right-hand side) indicated by a dense layer of droplets within the area of interest (where the two states are distinguishable) indicated by the white boxes.

A Panasonic DMC-FZ72 camera is permanently installed on a tripod and hung in a scaffold around the extraction column. The camera's image resolution is about 0.13 mm/pixel. Images are taken under various different illuminations, light from the right-, the left-hand side, both sides, or daylight only. An image preprocessing routine follows, where the desired image section is cut out as indicated in the right-hand side images in Fig. 2.

2.3 Modular Automation

For the implementation of ML scripts, following hardware and software architecture is used. Sensor data from the SBDC is transferred to and stored on the OPC UA server of the PLC. Current and historical values for each PEA can be accessed by the POL via the MTP interface. The MTP can be seen as part of the digital twin, describing the available architecture of sensors, actuators and functionalities of a PEA [53]. In order to implement ML methods to design an early flooding warning system, the sensor data is directly taken from the OPC UA servers via a Python script and fed to the developed monitoring system to determine the current operating state of the distillation column by classification. The monitoring concept is implemented in four steps: i) pressure drop preprocessing and filtering, ii) forecast by supervised learning methods, iii) operating state classification through clustering, iv) graphical output for the operator with forecast and classification. Data flow and implementation of the flooding prevention system are visualized in Fig. 3.

As described before, the distillation PEA is implemented to the POL via an open standard, the MTP. This architecture can be used additionally to implement the ML in a faster way, as the OPC UA server and needed process variables already exist and the location is completely described in the MTP file. Like this, the ML script is based on the existing architecture as an add-on for monitoring and classification of the set operation point. Furthermore, it is conceivable to implement a feedback loop as control for the distillation column. The ML script is then acting as a controller of the PEA. For the compliance with the MTP concept, the service of the distillation PEA needs to be adjusted with a new procedure. Within this new procedure, the spinning band speed is not anymore given by the POL, but self-adjusted by the PEA logic with the ML script as an internally acting controller within the distillation PEA logic. How-

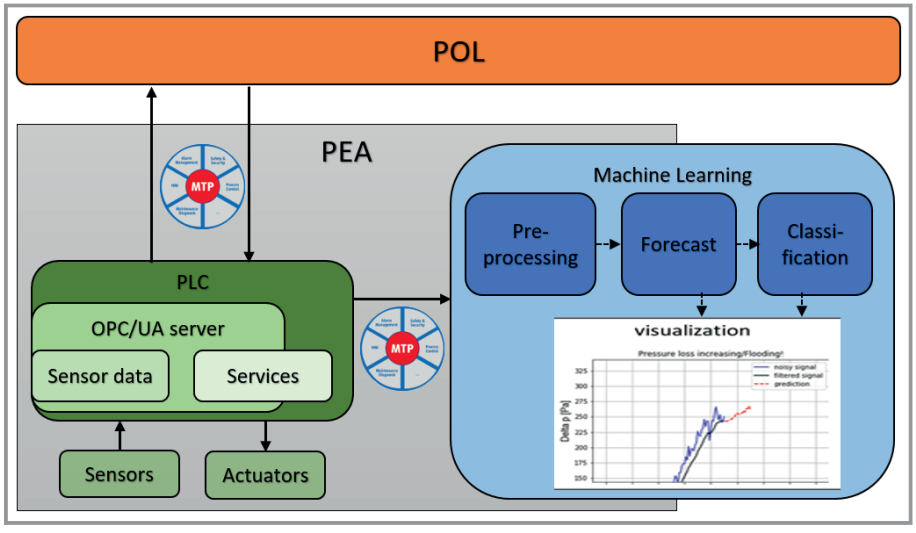

Figure 3. Software and data architecture of distillation PEA with integrated ML.

ever, this is not implemented yet, since the scope was to ensure the applicability of the forecast and classification first.

3 Flooding Detection in the Distillation Column by ML Tools

Generally, AI-supported tools for process monitoring and detection of malfunctions in process industry is of common interest and is becoming increasingly important in the context of industry 4.0 [54, 55]. Once the acceptance of such tools is given and operators are properly trained in its use, the benefit regarding resource, energy and time savings are huge. A lot of AI tools in context with downstream processing units found in literature are derived for industrial columns [56–58]. However, with a certain degree of instrumentation such models can already be implemented for laboratory or technicum columns in order to test the algorithms and tools in an early stage of process design. In the following sections, the preprocessing of the acquired sensor data, feature extraction, training of the ML methods and the implementation with live data are described.

3.1 Data Set and Preprocessing of Time Series Data

Time series from previous distillation column laboratory tests were used as training and test data. About 17 h of data were used, during which flooding occurred approx. 35 times within the column. For later validation, live data was used. The Python script accessed the OPC UA server of the control unit and processed it in real time.

Multivariate time series data can be tricky to deal with as the temporal structure should be preserved in some way during the training process. One way to make supervised learning methods applicable to time series data is the sliding window method [40], which transforms data in such a way that past and "future" measurements are preserved for each data point. For this use case, the future pressure drop (Δp_{t+1}, Δp_{t+2}, …) will be predicted based on the past data of pressure drop and other significant parameters X_i (…, Δp_{t-1}, $X_{1,t-1}$, … , Δp_t, $X_{1,t}$, …). These other significant parameters are determined in Sect. 3.2. A schematic representation of the sliding window data transformation is given in Fig. 4. The window size refers to the time window of past data and the response size describes the forecast window.

This transformation results in many additional data columns, which must be considered for the choice of an appropriate ML model. For the denoising of the pressure drop signal, an exponentially weighted moving average (EWMA) is used that weighs the most recent measurements stronger as they are more important to detect changes in trend and level of the pressure drop. For some ML methods that are based on distance (SVM, clustering), an additional scaling step is necessary that normalizes the input data. Linear regression and decision tree regressors do not require this scaling step.

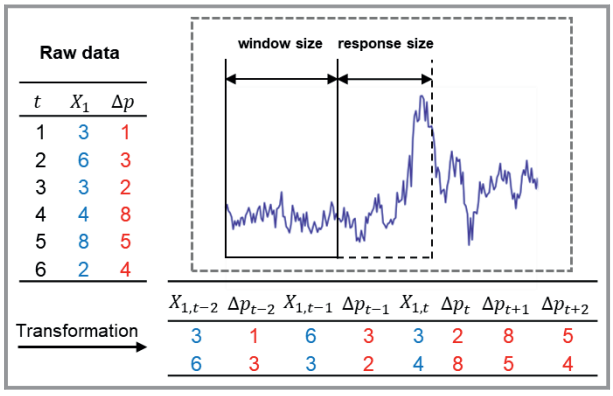

Figure 4. Sliding window method for the transformation of time series data.

3.2 Feature Extraction via Machine Learning

The spinning band distillation column contains the sensors and control variables summarized in Tab. 2.

Table 2. Sensors and control inputs of spinning band distillation column relevant for ML.

Sensor	Control variable
Pressure drop	Band rotation speed
Liquid level (bottom)	Heater power
Temperature (bottom)	Feed flow
Temperature (middle)	Reflux ratio
Temperature (head)	
Distillate mass	
Bottom product mass	

In order to train a robust and reliable ML model and avoid issues regarding co-linearity [2], it is helpful to pick significant features for the training process. This can be achieved by applying PCA or PLS regression, but since the number of parameters is manageable for this work, features are picked based on operator experience and a correlation matrix instead (Fig. 5). An advantage of this approach is that the results will be more interpretable compared to PCA or PLS regression, which addresses possible veracity issues as well.

The values in a correlation matrix describe the linear relationship between two parameters, where 1 and –1 indicate a perfectly linear relationship. The sign of the correlation coefficient describes the direction of the relationship: a positive sign means that the value of one parameter increases or decreases, if the other parameter increases or decreases, respectively; a negative sign describes the opposite relationship.

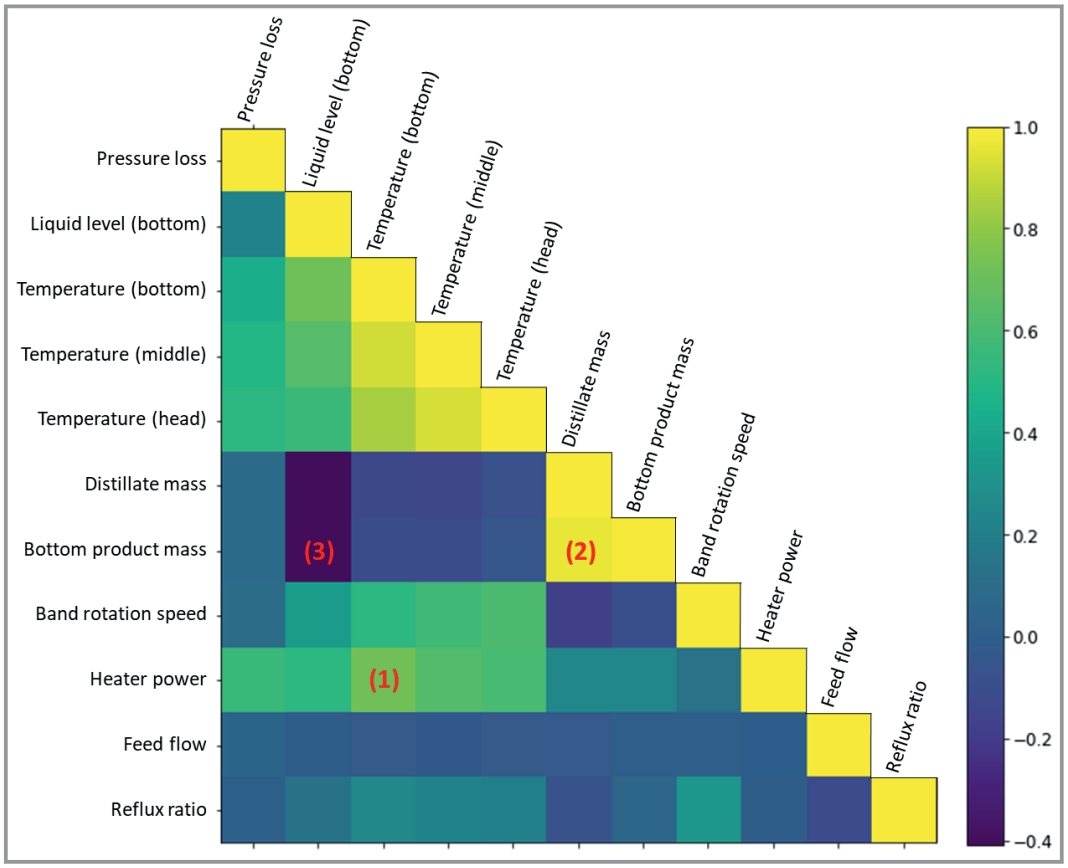

Figure 5. Correlation matrix for data of spinning band distillation column, left hand parameter has a linear impact to top hand parameter with positive or negative correlation.

The correlation matrix indicates that there is a strong linear relationship between the temperature measurements and heater power, which is expected for a distillation column (1). Further, there is a strong relationship between distillate and bottom product mass (2), but these features are not considered for the pressure drop forecast as it is known from experience that there is no significant impact on the pressure drop in the column. The same argument applies to the liquid level in the bottom (3). In terms of temperature measurements, the temperature in the head of the column is retained as a feature because it contains information on the boiling point of the volatile component and the current concentration. Pressure drop is kept as a feature as it describes the recent pressure drop trend, which can be useful for the forecast. The remaining parameters show no strong linear relationship. Furthermore, as known from experience and physical relationships the liquid holdup directly influences pressure drop in the distillation column. Thus, they are selected as features as well. In total, six parameters (pressure drop, column head temperature, band rotation speed, heater power, feed flow, and reflux ratio) are selected and used to model the forecast.

The clustering step will be performed based on the pressure drop data alone to identify flooding behavior in the distillation column. Pressure drop is preprocessed and transformed as described for the forecast problem in order to maintain the same data structure and facilitate the implementation with live data. Time series data can be typically decomposed into the following four features: trend, level, seasonality and noise. To ensure good visualization and interpretability of the occurring clusters, two features are chosen for the clustering process. As the flooding behavior does not occur in specific regular intervals (seasonality) and noise has been reduced by means of EWMA, trend and level should contain the significant information to identify meaningful clusters and are therefore chosen as features.

3.3 Model Training and Validation

Data from the spinning band distillation column is acquired in intervals of 1 s and since flooding happens abruptly, it is important to maintain this sample frequency despite the large amount of data that is collected. Therefore, scalable and computationally inexpensive models based on regression trees, which are explained in more detail in Sect. 1.1, are prioritized in the scope of this work. These bagging and boosting methods will be used with regression trees as base estimators for the pressure drop forecast and their performance will be compared based on chosen metrics, i.e., root

mean squared error and coefficient of determination (R^2). Additionally, linear regression will be applied for the pressure drop forecast to serve as a reference model.

The window and response size are determined preliminary via a grid search approach using a representative regression model (random forest regression). Investigated window sizes range from 5 to 20 s and response sizes from 15 to 30 s. The goal is to use a small window size to keep the amount of data during the transformation small (Fig. 3) and a large response size for a long forecast, while maintaining a good prediction accuracy on the test data set ($R^2 > 0.95$). Training data consists of 8 and test data of 2 recorded distillation runs, which corresponds to 54 948 and 9884 measurements, respectively. The resulting window and response size using this approach are identified as 10 s and 20 s, respectively. A shorter window size might result in missing the system's response to a change of control variables and larger response sizes lead to a strong decline in model accuracy.

For the pressure loss forecast the following ML methods are compared: linear regression, random forest, extra trees, AdaBoost, and gradient boosting regression. To achieve the best performance, the hyperparameters (e.g., number of estimators and depth of the decision trees) were optimized for the bagging and boosting regressors in a k-fold cross-validated grid search ($k = 5$) utilizing the training data. Finally, the performance of every model is compared in Tab. 3 based on the test data set and the chosen metrics. Note that AdaBoost and the extra trees regressor are combined via a voting regressor for an additional model as their training time is quite low. Due to the different working principles, weaknesses of the respective models could be eliminated by combining them. The training time is based on an INTEL Core i5-6600K CPU overclocked to 4.5 GHz.

Extra trees, gradient boosting, and the combination of AdaBoost and extra trees regression perform very similarly in terms of accuracy. If an adaptive ML solution is desired, the training time could play a critical role, but for the scope of this work all three highlighted models will be further evaluated with live data in the tryout stage (Sect. 3.5).

Table 3. Accuracy of pressure drop forecast for different algorithm methods based on the test data set.

Algorithm	RMSE	R^2	Training time [s]
Linear regression	12.7836	0.9361	0.38
Random forest	9.6330	0.9670	59.94
Extra trees	9.2411	0.9698	7.75
AdaBoost	10.1047	0.9636	37.85
Gradient boosting	9.0973	0.9700	275.08
AdaBoost + Extra trees	9.0849	0.9703	110.99

For the monitoring system, the pressure drop time series will be classified based on the 20 s forecast window and the measurements from the last 10 s to determine the current operating state of the distillation column, i.e., flooding or not flooding. Therefore, the historical pressure drop data is preprocessed as described in Sect. 3.1 with a window size of 30 s and used to identify meaningful clusters (k-means clustering) and label the data. As this method is distance based, a prior scaling of the data is performed. The time series features trend and level are determined from the slope of a linear fit and the median of the 30 s windows, respectively. Additionally, the data is filtered for positive slopes and medians above 80 Pa as flooding is only observed for increasing pressure drop trends and high pressure drop levels.

In order to identify an appropriate number of clusters, the elbow method is applied [59]. For this method, the k-means clustering algorithm is executed with a varying number of clusters, which are plotted against the inertia, i.e., the sum of squared distances of samples to their closest cluster center. Adding additional cluster centers after there are already enough clusters to describe the data, leads to a smaller change in inertia and a characteristic kink (elbow) is observed in the plot (Fig. 6, left). For the presented data this elbow is found at five clusters. In Fig. 6 (right) the data and cluster centers are visualized by a median against slope plot. Most of the data lies around a slope of 0, as it is the

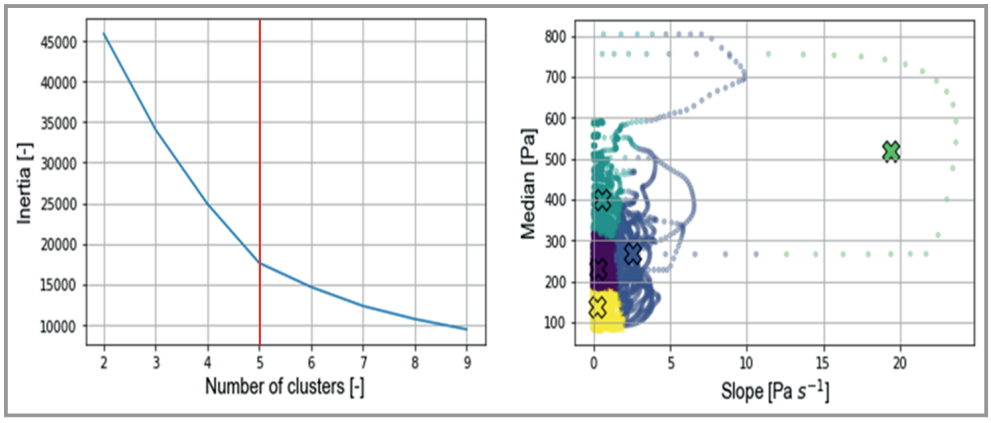

Figure 6. Elbow method for choosing the number of clusters with k-means clustering (left); median and slope plot with resulting clusters (right).

desired stable operating state. The blue and green (far right-hand side) clusters describe operating states, where the pressure drop is increasing, and the column might flood soon. For the implementation with live data a warning will be displayed if those conditions are observed. The remaining clusters will indicate normal operating behavior. Note that the data was transformed back to the original values for the visualization, but scaled data was used for the clustering process.

3.4 Implementation

As shown in Fig. 3, the distillation PEA is equipped with a PLC as a PEA internal control unit. An OPC UA server is located on this PLC, on which all process variables are published relevant for the process. The modular concept with OPC UA provides a standardized server interface, on which the ML algorithm can be adapted. The ML tool developed in Python has an OPC UA client taken from the Python package freeopcua [60], which reads the process variables every second and processes them into a data frame. This data frame can be processed further as shown above. Therefore, the data is preprocessed according to Sect. 3.1 and the developed models are applied to the data frame. As output the models provide on the one hand the prediction of the pressure for the next 20 s. On the other hand, the current operating status is classified from the prediction. The structure of the ML forecast implementation as well as the results plotted in a real time diagram are shown in Fig. 3.

The diagram shows the curve of the pressure difference, the filtered pressure curve and the prediction of future pressure difference. The current operating status is displayed above the diagram as text, which informs the operator if flooding occurs.

3.5 Distillation experiments and optimization

As validation data, a test procedure is carried out, in which the column is flooded several times. Care is taken to ensure that the flooding is generated by various parameter changes in order to check whether the influence of all parameters on the flooding behavior is reliably mapped. It could be shown that all three trained algorithms (gradient boost, extra trees, AdaBoost + extra trees) are able to detect and reliably display the flooding behavior. The accuracies (coefficient of determination) of the different models with respect to the validation data are $R^2_{\text{gradient boost}} = 0.878$, $R^2_{\text{extra trees}} = 0.853$ and $R^2_{\text{AdaBoost + extra trees}} = 0.857$.

The results of the flooding detection for the trained and selected models are shown in Fig. 7. It can be seen that all three models have similar accuracies, but the resulting prediction curves show significant differences. The prediction of the combined model of AdaBoost and extra trees regression shows a strongly fluctuating behavior, which makes it difficult to evaluate the forecast. The pure regression by extra trees is much smoother, but the prediction is too flat and has problems to follow the current pressure curve. In contrast, the Gradient Boost model shows a significantly more reactivated response with sufficient smoothing of the prediction curve, which makes this model the most suitable solution out of the three models investigated.

After the gradient boost algorithm is selected as the most suitable one, the response of the model will be discussed in detail. The data that the model receives as input is generated specifically for evaluation and has not been exposed to the model at any previous time. Fig.8 shows the response of the model to changes in the speed of the spinning band. At the beginning a steady state is shown, which is expressed by a filtered pressure value of about 190 Pa. The spinning band speed is set to a constant value of 300 rpm. This results in an

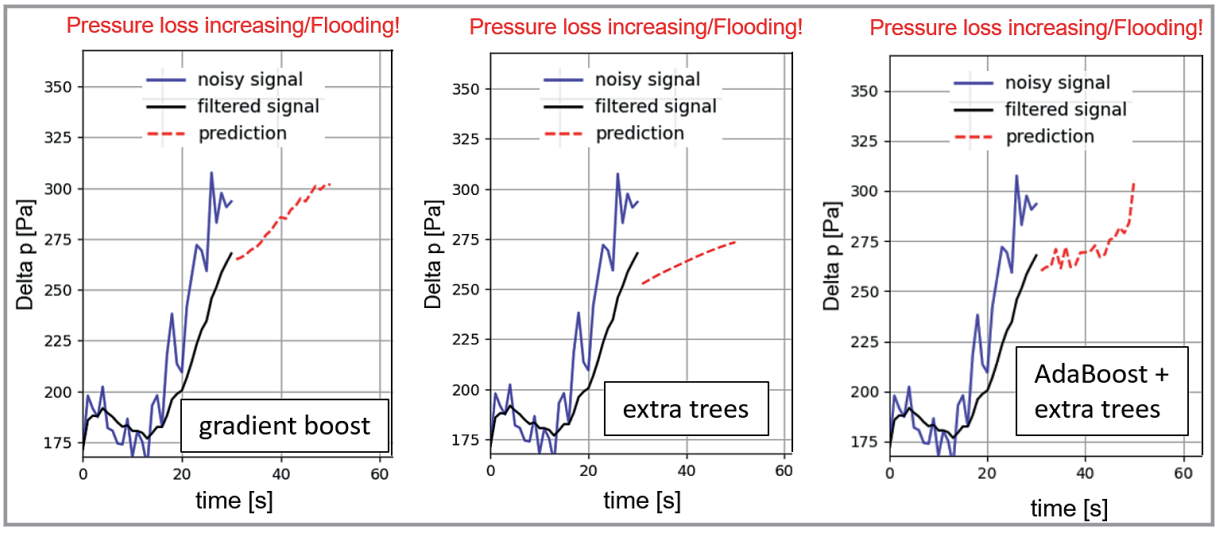

Figure 7. Behavior of the model response different trained models while flooding occurs in the spinning band distillation column.

almost stable prediction of the pressure difference. The classification on the top of the diagram shows that the column is working in a normal operating window (see Fig. 8, left).

Subsequently, after approx. 30 s, the speed of the spinning band is increased from 300 rpm to 400 rpm (see Fig. 8, right). This results in an increase of the pressure difference and the forecast predicts a stronger rising trend for the next 20 s. Due to the existing learned correlations, the classification of the model indicates already at this early stage that the column is entering a flood point, which enables the operator to take appropriate countermeasures.

It can thus be shown that the ML model is able to clearly predict the flooding of the SBDC, therefore in the following it will be analyzed in more detail what happens when the speed of the spinning band is reduced (see Fig. 9).

The speed is reduced to 150 rpm after 60 s. As a result of inertia within the system such as the slow discharge of the liquid in the flooding area of the column, there is a delayed flattening of the pressure difference. At this point, the course of the model deviates from the actual course, as a falling trend for the next 20 s is already apparent in the model. The classification also indicates that a stable operating condition will be achieved. This is due to the fact that the model was strongly adapted to the speed of the spinning band as a decisive influencing parameter. After 110 s a stationary state is reached, which shows that the classification works reliably.

4 Liquid-Liquid Extraction Column Flooding Detection

The images are labeled according to the operating state "normal operating state" or "flooding". 1344 images per class are fed as training data to the neural net. This data set is subdivided into 80 % training data and 20 % validation data. An external test data set with 252 images per class is provided as well. Image size is 224×224 pixel, with a few pixel of deviation regarding the shown image section that have no influence whatsoever on the classification results.

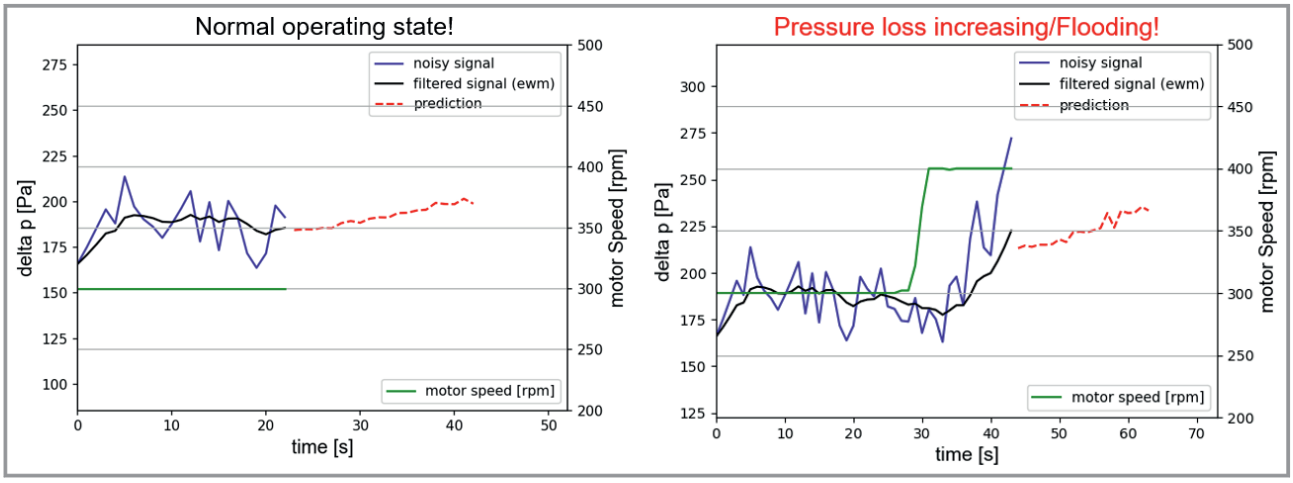

Figure 8. Response of the model in steady state (left) and an increase of the spinning band rotational speed (right).

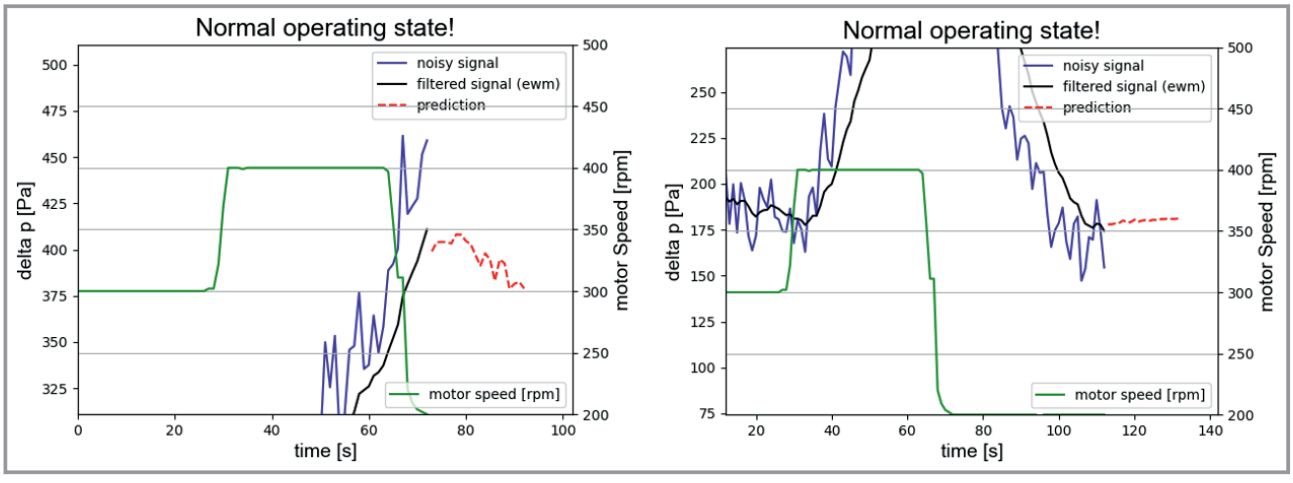

Figure 9. Response of the model to a decrease of the spinning band rotational speed.

As convolutional neural network CNN resnet18 is used [61] and retrained for this purpose. The core idea of ResNet is introducing a so-called "identity shortcut connection" that skips one or more layers. Since the shortcut connection is learning only the residual, the whole module is called residual module. The shortcut connection's skipping of certain layers speeds up the training process of the net. Resnet-18 consists of 18 convolution blocks, which each consist of several different layers.

After seven training epochs the neural net achieves an accuracy of 99.3 % in recognizing the correct operating state of the training set. For training purpose, the batch size was chosen to 4 with stochastic gradient descent and momentum (SGDM) as solver and with an initial learning rate of 0.001.

To check whether the net provides a reasonable performance, a confusion matrix is created. Here, the predicted class of the network is compared to the true class that is given to the image. If the predicted and the true class are the same, the network can make correct predictions. For validation of the trained net a set of 252 test images with 126 for each state, not used for the training of the neural net beforehand, is used. The obtained accuracy is 99.7 % with a single misclassified image predicted as flooding state instead of regular operation state.

During the investigations, the following question came up: what if the net can predict the class correctly, but is based on unreasonable sections within the image? To exclude this error source from training a network, a class activation map (CAM) is introduced. Its purpose is to visualize, within which area of the image the neural net deems most important to base on its class prediction decision. Thus, CAMs support making the ML algorithm's decisions explainable. This CAM is constructed in Matlab by using the "activations()" function and plotting it on top of the analyzed image (see Fig. 10).

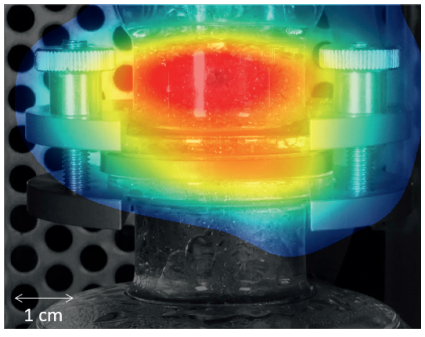

Figure 10. Class activation map of a flooding image. The neural net's main focus is on the red area with decreasing importance towards the blue area. Gray areas are identified as unimportant for the classification problem. Here, the upper part of the flange is taken as most interesting area for the decision.

It proves that the neural net focuses on a reasonable section within the image to determine whether it sees an image of the normal operating state or flooding. With these two criteria satisfied, a first step towards an image-based control of an extraction column is made.

Since the successful detection of the columns state using ML could be shown, the ML is now to be developed into an ML-based smart sensor for online monitoring, displaying the current state of the column. As an even further outlook, a feedback control tool is to be developed of this smart sensor. The AD Labs DN32 extraction column is built as a modular column according to VDI 2776, just like the spinning band distillation column. For automation it as well uses existing control architectures such as the MTP (cf. Fig. 3). Thus, the linkage of the results of this optical sensor to control the PEA system to for example increase pulsation or decrease stirrer speed to avoid flooding is the next feasible step.

5 Conclusion and Outlook

The application of machine and deep learning in the process industry is an adequate way to predict or detect the flooding behavior or malfunction of columns. In this case, flooding detection with ML tools were implemented to a laboratory distillation and liquid extraction column. It was shown that both time-series data of process values and image recognition can be used for modeling. Parallel to this, examples were given, which enable the simple integration of AI-based monitoring systems into existing plants enabled by existing control architectures such as the module type package. Adaptation of process parameters avoid flooding, reaction on disturbances, e.g., trace elements and their influence on the surface tension or surface wetting phenomena. An adaptation of camera setups or the existing data structures, such as OPC UA, are sufficient to provide an interface for a data science implementation. This results in a high potential for tooling up existing equipment with AI methods as part of the digital twin. It is possible to combine both analytical methods in order to specify the flooding behavior even more and to transfer the flooding detection from an AI-supported to an AI-controlled monitoring system.

The BMWi is acknowledged for funding this research in the ORCA project as part of the ENPRO2.0 initiative (Support code: 03ET1517B). The BMWi is acknowledged for funding this KEEN project initiative (Support code: 01MK20014S). Open access funding enabled and organized by Projekt DEAL.

Symbols used

R^2	[–]	coefficient of determination
RMSE	[–]	root mean squared error

Abbreviations

CAM	class activation map
CNN	convolutional neural network
DOE	design of experiments
EWMA	exponential weighted moving average
GBR	gradient boosting regressor
HMI	human machine interface
LSTM	long short-term memory
ML	machine learning
MTP	module type package
PCA	principal component analysis
PEA	process equipment assembly
PLC	programmable logical controller
PLS	partial least squares
POL	process orchestration layer
SBDC	spinning band distillation column
SVM	support vector machine

References

[1] B. Li, B. Hou, W. Yu, X. Lu, C. Yang, *Front. Inf. Technol. Electron. Eng.* **2017**, *18 (1)*, 86–96. DOI: https://doi.org/10.1631/FITEE.1601885

[2] P. Kadlec, B. Gabrys, S. Strandt, *Comput. Chem. Eng.* **2009**, *33 (4)*, 795–814. DOI: https://doi.org/10.1016/j.compchemeng.2008.12.012

[3] N. Mahony, T. Murphy, K. Panduru, D. Riordan, J. Walsh, *Machine Learning Algorithms for Process Analytical Technology*, IEEE, Piscataway, NJ **2016**.

[4] N. Kockmann, *React. Chem. Eng.* **2019**, *4 (9)*, 1522–1529. DOI: https://doi.org/10.1039/C9RE00017H

[5] S. L. Brunton, B. R. Noack, P. Koumoutsakos, *Annu. Rev. Fluid Mech.* **2020**, *52 (1)*, 477–508. DOI: https://doi.org/10.1146/annurev-fluid-010719-060214

[6] Q. Min, Y. Lu, Z. Liu, C. Su, B. Wang, *Int. J Inf. Manage.* **2019**, *49*, 502–519. DOI: https://doi.org/10.1016/j.ijinfomgt.2019.05.020

[7] Z. Ge, Z. Song, S. X. Ding, B. Huang, *IEEE Access* **2017**, *5*, 20590–20616. DOI: https://doi.org/10.1109/ACCESS.2017.2756872

[8] C. Cortes, V. Vapnik, *Mach. Learn.* **1995**, *20 (3)*, 273–297. DOI: https://doi.org/10.1007/BF00994018

[9] T. K. Ho, *Random Decision Forrests*, IEEE Computer Society Press, Los Alamitos, CA **1995**.

[10] T. Hastie, S. Rosset, J. Zhu, H. Zou, *Stat. Its Interface* **2009**, *2 (3)*, 349–360. DOI: https://doi.org/10.4310/SII.2009.v2.n3.a8

[11] L. Mason, J. Baxter, P. L. Bartlett, M. R. Frean, in *Proc. of Advance in Neural Information Processing Systems*, MIT Press, Cambridge, MA **1999**, 512–518.

[12] P. Kadlec, R. Grbić, B. Gabrys, *Comput. Chem. Eng.* **2011**, *35 (1)*, 1–24. DOI: https://doi.org/10.1016/j.compchemeng.2010.07.034

[13] P. Geurts, D. Ernst, L. Wehenkel, *Mach. Learn.* **2006**, *63 (1)*, 3–42. DOI: https://doi.org/10.1007/s10994-006-6226-1

[14] H. Drucker, C. Cortes, L. D. Jackel, Y. LeCun, V. Vapnik, Boosting and Other Machine Learning Algorithms, in *Machine Learning Proceedings 1994* (Eds: W. W. Cohen, H. Hirsh), Elsevier, Amsterdam **1994**. DOI: https://doi.org/10.1016/B978-1-55860-335-6.50015-5

[15] L. Breiman, *Mach. Learn.* **2001**, *45 (1)*, 5–32. DOI: https://doi.org/10.1023/A:1010933404324

[16] J. H. Friedman, *Comput. Stat. Data Anal.* **1999**, *38*, 367–378.

[17] M. M. Martín, *Introduction to software for chemical engineers*, CRC Press, Boca Raton, FL **2019**.

[18] D.-H. Kim, T. J. Y. Kim, X. Wang, M. Kim, Y.-J. Quan, J. W. Oh, S.-H. Min, H. Kim, B. Bhandari, I. Yang, S.-H. Ahn, *Int. J. Precis. Eng. Manuf.-Green Technol.* **2018**, *5 (4)*, 555–568. DOI: https://doi.org/10.1007/s40684-018-0057-y

[19] H. Zheng, Y. Wu, *Appl. Sci.* **2019**, *9 (15)*, 3019. DOI: https://doi.org/10.3390/app9153019

[20] S. J. Qin, L. H. Chiang, *Comput. Chem. Eng.* **2019**, *126*, 465–473. DOI: https://doi.org/10.1016/j.compchemeng.2019.04.003

[21] C. E. Rasmussen, C. K. I. Williams, *Gaussian processes for machine learning*, 3rd ed., Adaptive computation and machine learning, MIT Press, Cambridge, MA **2008**.

[22] J. Schmidhuber, *Neural Networks* **2015**, *61*, 85–117. DOI: https://doi.org/10.1016/j.neunet.2014.09.003

[23] A. Diez-Olivan, J. Del Ser, D. Galar, B. Sierra, *Inf. Fusion* **2019**, *50*, 92–111. DOI: https://doi.org/10.1016/j.inffus.2018.10.005

[24] L. Chiang, B. Lu, I. Castillo, *Annu. Rev. Chem. Biomol. Eng.* **2017**, *8*, 63–85. DOI: https://doi.org/10.1146/annurev-chembioeng-060816-101555

[25] V. Venkatasubramanian, *AIChE J.* **2019**, *65 (2)*, 466–478. DOI: https://doi.org/10.1002/aic.16489

[26] S. S. Kamble, A. Gunasekaran, S. A. Gawankar, *Process Saf. Environ. Prot.* **2018**, *117*, 408–425. DOI: https://doi.org/10.1016/j.psep.2018.05.009

[27] M. W. Hlawitschka, J. Schulz, D. Wirz, J. Schäfer, A. Keller, H.-J. Bart, *Chem. Ing. Tech.* **2020**, *92 (7)*, 914–925. DOI: https://doi.org/10.1002/cite.202000043

[28] N. Kockmann, L. Bittorf, W. Krieger, F. Reichmann, M. Schmalenberg, S. Soboll, *Chem. Ing. Tech.* **2018**, *90 (11)*, 1806–1822. DOI: https://doi.org/10.1002/cite.201800020

[29] B. Sun, S.-L. Jämsä-Jounela, Y. Todorov, L. E. Olivier, I. K. Craig, *IFAC-PapersOnLine* **2017**, *50 (2)*, 65–70. DOI: https://doi.org/10.1016/j.ifacol.2017.12.012

[30] VDI/VDE/NAMUR 2658 Part 1, *Automation engineering of modular systems in the process industry – general concept and interfaces*, VDI, Düsseldorf **2019**.

[31] VDI/VDE/NAMUR 2658 Part 2, *Automation engineering of modular systems in the process industry – Modelling of human-machine-interfaces*, VDI, Düsseldorf **2019**.

[32] VDI/VDE/NAMUR 2658 Part 3, *Automation engineering of modular systems in the process industry – library for data objects*, VDI, Düsseldorf **2019**.

[33] VDI/VDE/NAMUR 2658 Part 4, *Automation engineering of modular systems in the process industry – modelling of module services*, VDI, Düsseldorf **2020**.

[34] VDI 2776 Part 1, *Modular Plants – General concept and design of modular plants*, VDI, Düsseldorf **2019**.

[35] A. Klose, S. Merkelbach, A. Menschner, S. Hensel, S. Heinze, L. Bittorf, N. Kockmann, C. Schäfer, S. Szmais, M. Eckert, T. Rüde, T. Scherwietes, P. da Silva Santos, F. Stenger, T. Holm, W. Welscher, N. Krink, T. Schenk, A. Stutz, M. Maurmaier, K. Stark, M. Hoernicke, S. Unland, S. Erben, F. Kessler, F. Apitz, L. Urbas, *Chem. Eng. Technol.* **2019**, *42 (11)*, 2282–2291. DOI: https://doi.org/10.1002/ceat.201900298

[36] H. Bloch, S. Hensel, M. Hoernicke, K. Stark, A. Menschner, A. Fay, L. Urbas, T. Knohl, J. Bernshausen, *Procedia CIRP* **2018**, *72*, 1088–1093. DOI: https://doi.org/10.1016/j.procir.2018.03.037

[37] B. Cao, L. A. Adutwum, A. O. Oliynyk, E. J. Luber, B. C. Olsen, A. Mar, J. M. Buriak, *ACS Nano* **2018**, *12 (8)*, 7434–7444. DOI: https://doi.org/10.1021/acsnano.8b04726

[38] J. A. Carvajal Soto, F. Tavakolizadeh, D. Gyulai, *Int. J. Comput. Integr. Manuf.* **2019**, *32 (4–5)*, 452–465. DOI: https://doi.org/10.1080/0951192X.2019.1571238

[39] K. E. Murray, *J. Am. Oil Chem. Soc.* **1951**, *28 (6)*, 235–239. DOI: https://doi.org/10.1007/BF02678899
[40] L. Bittorf, N. Böttger, D. Neumann, A. Winter, N. Kockmann, *Chem. Eng. Technol.* **2021**, *44 (9)*, 1660–1667. DOI: https://doi.org/10.1002/ceat.202000602
[41] L. Bittorf, M. Schrimpf, N. Böttger, N. Kockmann, *Chem. Ing. Tech.* **2018**, *90 (9)*, 1313. DOI: https://doi.org/10.1002/cite.201855391
[42] H. Z. Kister, in *Distillation: Operation and Application* (Eds: A. Górak, H. Schoenmakers), Elsevier, Amsterdam **2014**.
[43] H. Z. Kister, *Distillation Design*, McGraw-Hill, New York **1992**.
[44] L. Bittorf, N. Kockmann, P. S. Da Santos, K. Stark, M. Hoernicke, T. Holm, A. Stutz, M. Eckert, A. Menschner, A. Klose, S. Merkelbach, L. Urbas, in *Automation 2020 Conference*, VDI-Verlag, Düsseldorf **2020**, 129–144. DOI: https://doi.org/10.51202/9783181023754-129
[45] P. J. King, B. J. Yates, *Chem. Process Eng.* **1966**, *47 (5)*, 214–223.
[46] *Distillation: Fundamentals and Principles* (Eds: A. Górak, E. Sorensen), Elsevier, Amsterdam **2014**.
[47] L. Bittorf, J. Oeing, T. Holm, S. Loepker, M. Hoernicke, K. Stark, N. Kockmann, *Service oriented architecture for an automated laboratory distillation process module*, Prozess-, Apparate-, und Anlagentechnik PAAT, Dortmund **2019**.
[48] S. Soboll, I. Hagemann, N. Kockmann, *Chem. Ing. Tech.* **2017**, *89 (12)*, 1611–1618. DOI: https://doi.org/10.1002/cite.201700031
[49] A. Rathgeb, A. Palmtag, S. Kaminski, A. Jupke, *Chem. Ing. Tech.* **2019**, *91 (12)*, 1766–1776. DOI: https://doi.org/10.1002/cite.201900163
[50] S. Ousmane, M. Isabelle, M.-S. Mario, T. Mamadou, A. Jacques, *Chem. Eng. Res. Des.* **2011**, *89 (1)*, 60–68. DOI: https://doi.org/10.1016/j.cherd.2010.04.011
[51] R. P. Panckow, L. Reinecke, M. C. Cuellar, S. Maaß, *Oil Gas Sci. Technol. – Rev. IFP Energies nouvelles* **2017**, *72 (3)*, 14. DOI: https://doi.org/10.2516/ogst/2017009
[52] J. Villwock, *Systematische Analyse des Koaleszenzverhaltens von zweiphasigen Flüssigsystemen bei Ionenzugabe*, Dissertation, Technische Universität Berlin **2019**.
[53] A. Bamberg, L. Urbas, S. Bröcker, M. Bortz, N. Kockmann, *Chem. Eng. Technol.* **2021**, *44 (6)*, 954–961. DOI: https://doi.org/10.1002/ceat.202000562
[54] K. Severson, P. Chaiwatanodom, R. D. Braatz, *Annu. Rev. Control* **2016**, *42*, 190–200. DOI: https://doi.org/10.1016/j.arcontrol.2016.09.001
[55] *Handbuch Industrie 4.0* (Eds: B. Vogel-Heuser, T. Bauernhansl, M. ten Hompel), Springer, Berlin **2016**.
[56] P. Omoarebun, D. Sanders, M. Haddad, M. Hassan, G. Tewkesbury, K. Giasin, in *2020 IEEE 10th International Conference on Intelligent Systems (IS): Proceedings* (Eds: V. Sgurev), IEEE, Piscataway, NJ **2020**.
[57] J. Picabea, M. Maestri, M. Cassanello, G. Horowitz, *Chem. Prod. Process Model.* **2020**, *16 (3)*, 169–180. DOI: https://doi.org/10.1515/cppm-2020-0004
[58] S. A. Taqvi, L. D. Tufa, H. Zabiri, S. Mahadzir, A. Shah Maulud, F. Uddin, in *Modeling, Design and Simulation of Systems* (Eds: M. S. Mohamed Ali et al.), Vol. 751, Communications in Computer and Information Science, Springer Nature, Singapore **2017**.
[59] T. Kodinariya, P. R. Makwana, *Int. J. Adv. Res. Comput. Sci. Manage. Stud.* **2013**, *1*, 90–95.
[60] O. Roulet-Dubonnent, *Python OPC UA Documentation – Release 1.0*, **2018**.
[61] K. He, X. Zhang, S. Ren, J. Sun, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Piscataway, NJ **2016**, 770–778. DOI: https://doi.org/10.1109/CVPR.2016.90
[62] NE 175, *NAMUR Open Architecture – NOA Concept*, NAMUR – Interessengemeinschaft Automatisierungstechnik der Prozessindustrie e.V., Leverkusen **2020**.
[63] L. Bittorf, et al., *Chem. Eng. Technol.* **2021**, *44 (9)*, 1660–1667. DOI: 10.1002/ceat.202000602
[64] L. Bittorf, *Ind. Eng. Chem. Res.* **2021**, *60 (30)*, 10854–10862. DOI: 10.1021/acs.iecr.1c01326

