Digital Chemical Engineering 18 (2026) 100289

Contents lists available at ScienceDirect

Digital Chemical Engineering

journal homepage: www.elsevier.com/locate/dche

Original article

Kolmogorov–Arnold and deep learning networks for industrial explainable product quality prediction

Balázs Fricz [a], Gergely Horváth [a,b], Alex Kummer [a,c,*]

[a] Department of Process Engineering, University of Pannonia, Egyetem st. 10, H-8200 Veszprem, Hungary
[b] BorsodChem Zrt. 1 Bolyai Square, Kazincbarcika H-3700, Hungary
[c] HUN-REN-PE Complex Systems Monitoring Research Group, University of Pannonia, Egyetem st. 10, H-8200 Veszprem, Hungary

ARTICLE INFO

Keywords:
Soft sensor
Explainable machine learning
Kolmogorov–Arnold networks
Industrial quality estimation
Neural network

ABSTRACT

In the chemical process industry, the frequent measurement of quality variables is not always a viable path to take. It can be due to the measurement being time- or cost-consuming, leading to a sampling time of hours or even a day. As such, soft sensors were developed as a solution to this problem, providing a model trained on the available output measurements and predicting its value between samplings. This paper presents the process of soft sensor development for an industrial case study and the application of recently developed Kolmogorov–Arnold Networks (KAN) as a possible soft sensor model, where also the inherent interpretability of KAN through symbolic regression is analyzed.

The soft-sensor development and interpretability analysis is performed on a benchmark dataset (steam turbine dataset), and on a real industrial case study (aniline synthesis plant). The prediction performance of KAN is compared to different complexity ML models, and in addition, the models were compared on the basis of their interpretability and explainability using Shapley values. The results show that the inherent interpretability of KAN models using symbolic regression is too complicated on simple tasks, thus losing its ability to explain the model estimations, though its model performance is similar to the classic feedforward neural networks.

1. Introduction

In the current industrial environment, trading partners expect product qualities that meet contractual and regulatory requirements. Because off-spec material increases purification costs and waste, critical quality variables must be monitored and controlled. However, these variables are often measured infrequently because sampling and analysis are difficult, slow or expensive, so corrective actions may be delayed by hours, leading to large quantities of off-specification product (Fortuna et al., 2007).

Soft sensors can bridge these sampling gaps by predicting rarely measured quality variables from continuously collected process data (Lin et al., 2007). Beyond providing online estimates, they also promote systematic use of historical plant data for safer and more efficient operation (Lin et al., 2007; Graziani and Xibilia, 2020). In this work, we design a soft sensor for an industrial aniline reactor, following the full workflow from data collection to model validation while comparing several advanced modeling methods.

There are various examples of tasks solved using soft sensors. The process for the testing of the sensor is often a distillation column, in which either the distillate or the product composition is modeled (Kaneko et al., 2009; Kim et al., 2013). Besides this there are various other case studies employed for sensor testing such as wastewater treating (Wang et al., 2022; Yan et al., 2020), steel production (Feng et al., 2020), absorption (Liu et al., 2018), flotation (Chen et al., 2024) and polymer production (Wu et al., 2021).

As mentioned previously, the applied models for the soft sensors can range from linear methods based on principal component analysis (PCA) and partial least squares (PLS) fitting (Kaneko et al., 2009; Foschi et al., 2021) to a wide variety of non-linear machine learning based approaches. Some of the non-linear sensors use autoencoders (Yuan et al., 2018) or long-short term memory networks (Yuan et al., 2020), while others apply genetic programming (Ferreira et al., 2022) or graphs (Feng et al., 2020; Chen et al., 2024; Wu et al., 2021), but also many other variations of neural networks exist in the literature (Wang et al., 2022; Yan et al., 2020). There are modifications of these models that deal with the change of the process (e.g. catalyst deactivation) through adaptation. For example just-in-time methods can be effectively used for both linear models (Kim et al., 2013; Shao et al., 2015) and neural networks (Liu et al., 2018).

* Corresponding author at: HUN-REN-PE Complex Systems Monitoring Research Group, University of Pannonia, Egyetem st. 10, H-8200 Veszprem, Hungary.
E-mail address: kummer.alex@mk.uni-pannon.hu (A. Kummer).

https://doi.org/10.1016/j.dche.2026.100289
Received 9 September 2025; Received in revised form 30 December 2025; Accepted 17 January 2026
Available online 19 January 2026
2772-5081/© 2026 The Authors. Published by Elsevier Ltd on behalf of Institution of Chemical Engineers (IChemE). This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

The Kolmogorov–Arnold Network (KAN) is a method developed by Liu et al. (2024) and is meant to serve as an alternative to "standard" artificial neural networks (ANNs). Structure-wise it is similar to ANNs, but KANs employ splines for univariate estimation purposes. This structure is claimed to be able to represent complex nonlinear systems better. The authors of Liu et al. (2024) claim that KANs can provide comparable or even better results than standard ANNs, and as such it is a promising alternative that is worth investigating. As KAN is a recent development, in this paper we intended to expand the literature that applies the network to real life tasks, which in this case is the prediction of a by-product in an aniline synthesis reactor.

KAN has been used in different case studies for soft-sensing tasks. In Hu et al. (2025) KAN was compared with various time series predicting neural networks (LSTM, GRU etc.) to predict the output of an oil well, where its performance was similar to the other models. In another article KAN was applied to the modeling and optimization of the energy efficiency of buildings. In the same paper an additional case study of a thermal power plant was described, where the produced power, turbine heat rate and thermal efficiency were estimated and compared to neural networks. The results showed that KAN had a comparable predictive power to the neural networks (Ansar and Ashraf, 2025). An adapted version of KAN was used to predict cancer survival rate of patients in both synthetic and real clinical datasets. It was compared against the most often used survival regression model (CoxPH) and a deep neural network modified predictor. The results showed that KAN can achieve comparable accuracy as the other tested methods, in some cases outperforming them (Knottenbelt et al., 2024). KAN outperformed support vector regression, random forest regression and Gaussian process regression in a time series forecasting task, capturing trends and dynamic fluctuations with high accuracy (Saravani et al., 2025). In the estimation of a heat transfer parameter it performed slightly worse than support vector regression and classic neural networks with much higher computational time (Abramov et al., 2024). The reviewed articles often took advantage of the non-linear predictive capabilities and the fewer parameters or the potentially interpretable symbolic regression available in KAN.

However neither our review of the available literature nor the survey made by Somvanshi et al. (2024) or Ji et al. (2024) found any applications as a chemical industrial soft sensor, hence so far this is an unexplored area of research, and our intention is to assist in filling this gap. The symbolic capabilities of KAN might be of use when designing predictor models that are accurate and with careful training and pruning also interpretable by design, as this can help process engineers discover and exploit knowledge about the technology.

As chemical production systems and sensor networks grow in complexity, data-driven models for quality prediction increasingly rely solely on process data, without explicit physical constraints. This can obscure why a particular prediction is made and reduce operator trust in model-based recommendations (Faubel et al., 2023). Consequently, explainable or interpretable AI methods have gained importance, even though interpretability is not a strictly mathematical concept. There is a wide variety of definitions available in the literature (Carvalho et al., 2019; Doshi-Velez and Kim, 2017; Li et al., 2022; Yuan et al., 2024), and the basic idea in many of them is: make the system predictions human understandable or retraceable, either by design or through a supporting method. This interpretability can be global (the model itself is interpretable) or local (the predictions themselves are explained) (Kotriwala et al., 2021). The assessment of interpretability and interpretable model creation can be done at multiple point during the machine learning task. In the pre-modeling step, proper data analysis and feature selection can aid in constructing interpretability models. In-model interpretability is concerned with selecting model structures that are explainable by design, such as the polynomial neural network presented in Chaffart and Yuan (2025). After training (*post hoc*) interpretability relates to the examination of the explainability of the final model using various methods. These methods can be specific to a certain model type or independent of it (called model-agnostic) (Li et al., 2022; Carvalho et al., 2019).

The goal of our work was to investigate the performance and inherent interpretability of KAN models through two case studies: one based on a benchmark dataset of a steam turbine (Tüfekci, 2014), and another involving an industrial aniline synthesis reactor. The first case study assessed KAN's ability to generate symbolic equations and its SHAP-based interpretability compared to shallow and deep neural networks, using a benchmark steam turbine dataset. The second study applied KAN as a soft sensor in an industrial aniline synthesis process to estimate by-product concentration. Multiple models were trained on the same data with optimized hyperparameters and evaluated through standard modeling steps. SHAP analysis was used to compare the interpretability of KAN and baseline neural networks. The contributions of our work are the following:

- The inherent interpretability of the KAN using symbolic regression does not work for even simple regression tasks, as presented on the benchmark turbine dataset;
- Developing hyperparameter optimized KAN and neural network-based soft sensors for estimating the by-product concentration in a real industrial system;
- The model accuracy of KAN can reach the accuracy of classic feedforward neural networks, though on the analyzed case studies it did not present improvement compared to them;
- Shapley-based evaluation as post-explanation of the developed soft sensor models (shallow network, deep network, Kolmogorov–Arnold network) for learning about the prediction generation trends.

2. General methodology of soft sensor development

Soft-sensor development can be organized into four stages: data preparation (collection and preprocessing), variable selection, model design and validation, and finally sensor implementation and maintenance (sensor use), as summarized in Fig. 1. In this paper we focus on the stages up to and including model validation and explainability analysis; deployment and maintenance are process-specific and therefore not treated in detail.

The first step is the preparation of the data, starting with data collection. When collecting the samples, care must be taken to ensure that the statistical distributions of the modeling dataset are similar to the ones encountered during the standard operation of the process. This is necessary to ensure that edge cases do not have excessive influence on the accuracy of the trained sensor (Ben-David et al., 2010). There are also other issues and solutions, such as the presence of potentially useful inter-sample data (Lin et al., 2009; Rato and Reis, 2017), missing variables in data rows that need to be handled (Andridge and Little, 2010) and outlier data (Souza et al., 2016; Jiang et al., 2020).

The large amount of preprocessed data contains variables that vary in the level of exhibited noise and usefulness, so the next step is selecting the variables that will be used in the soft-sensor. This can be aided by the process expert's knowledge, however modern production systems are integrated and physically large, leading to potential hidden connections unseen until in-depth data analysis is carried out (Souza et al., 2016). The variable selection can include all variables (but this leads to high computational costs and a potential for overfitting), be done by hand, using wrappers like sequential selection (Aha and Bankert, 1995) or unsupervised methods like principal component analysis (PCA) (Song et al., 2010).

The soft sensor itself is a model of the process, connecting process inputs or state variables to the output. This model can be based on first-principle equations that rely on knowledge and understanding of the system to construct various balance equations based on physical–chemical laws (white-box models) (Bradley et al., 2022). The other end of this is when white-box equations cannot be constructed and

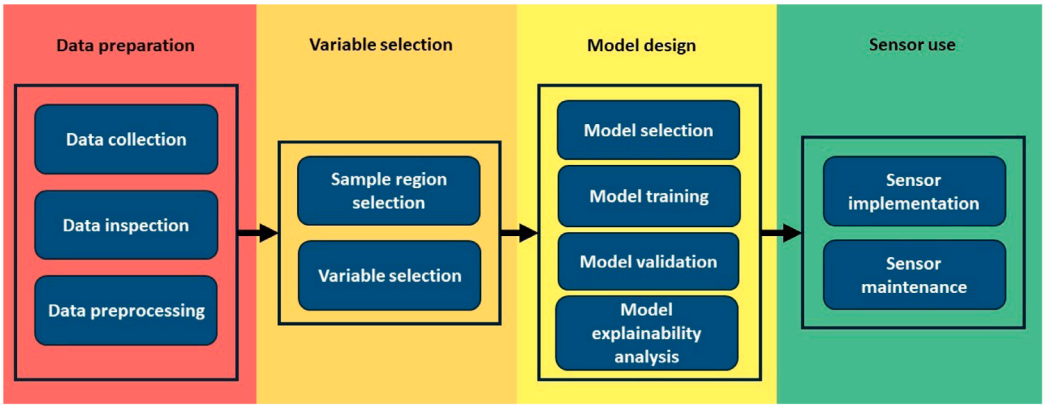

Fig. 1. Overview of the steps required to synthesize a soft sensor (figure made by the authors).

instead arbitrary equations are fitted on the samples, using process data (black-box models). White-box models can often be hard to construct due to the complexity of the technology, but can be used to analyze a variety of states (startup, shutdown, new setpoints) and for optimization (Nazemzadeh et al., 2021). Between these two edge cases there are gray-box models that are often first-principle equations with black-box parameter estimators or data-based models with constraints based on physical–chemical laws (e.g. physics informed neural networks) (Bradley et al., 2022).

During the sensor design the first decision is to choose whether to make a whitebox or blackbox model. If the blackbox is chosen, the selected variables need to be used in the regression model. There is a wide variety of models available for soft sensor applications, from linear methods like multiple linear regression (MLR), PCA based regression or partial least squares (PLS) to various non-linear ones like k-nearest neighbor (kNN) and neural networks (NN) and its various improvements (convolution, graph structures, recurrent networks etc.) (Souza et al., 2016). An important thing to note is that a linear model should always be tested as the sensor model, as it might be able to provide the necessary accuracy while retaining simplicity.

2.1. Feature engineering

Real industrial data can come from two different sources: from sensors that provide frequent measurements and from laboratory analysis, which is often infrequent due to the associated costs and the time consuming nature of such tests. However the source of modeling data does not provide information on its quality, both sources can have outliers and regions where data is missing. Thus the first steps are to remove the outliers, match the different sources of data and fill or remove the incomplete data rows.

The preprocessed data is then used for the selection of features that will be used in the synthesis of the model for the sensor. This requires identifying the subset of the input variables that are suitable for the given task. The measure of the suitability of the selected input variables (**X**) is their impact on the output variables (**Y**). One method of finding the **X** that affect **Y** the most is to check the correlation between the two variables.

Two significant issues can arise when developing a soft sensor. In ideal conditions a process operates at one or few steady state point(s), which means that a large available dataset might still be inadequate to provide the necessary coverage of the input variable space. This fact can hinder the predictive power of the designed soft sensor (Ferreira et al., 2022).

Another important issue present in chemical engineering is that the chemical process systems can have large residence times. It means that a change in **X** at time t will only affect **Y** at $t+\tau$ where τ is the time delay or mean residence time of the system. This necessitates the rigorous matching of the inlet variables with the proper outlet variables. It is not a simple task as it requires an estimation of the system's residence time (Ferreira et al., 2022).

The selection of variables for the development of a soft sensor can be done in multiple ways. All variables can be used in the model, or the knowledge of the system expert can aid the selection process. There are unsupervised (e.g. PCA or autoencoders) and supervised (e.g. based on Pearson correlation, sequential selection) techniques, each having advantages and disadvantages. In the current work the selection of the variables was done by hand, relying on the combination of Pearson correlations, visualization and the system expert's knowledge.

After the modeling variables are selected, the data is divided into training, testing and validation sections. As training is the most crucial step, the sample size of the training dataset need to be the largest. In our case 70% of the samples were used for training and 20% and 10% for testing and validation. The samples were selected randomly the data was standard scaled for use in the model.

2.2. Model structure selection and validation

For soft-sensing, a wide array of linear and non-linear methods is available. The prediction task can be generalized in the following form:

$$\hat{\mathbf{y}} = f(\mathbf{X}, \theta) \tag{1}$$

Here $\hat{\mathbf{y}}$ is the prediction vector for measured variable $\mathbf{y}$ with (N x 1) dimension, where N is the number of samples, $\mathbf{X}$ is the matrix of independent variables with (Nxd) dimension, where d is the number of independent variables. Function f represents the prediction model parameterized by θ. The prediction is evaluated using an error function, some of the commonly ones are the R^2 value, the mean square error (MSE) or its root (RMSE), mean absolute error (MAE) and the mean absolute percentage error (MAPE).

As many modern chemical industrial processes are highly nonlinear, in the current work, a multiple linear regression (MLR) model, a Support Vector Regression model, an XGBoost regression tree model, a shallow and a deep artificial neural network (ANN), and KAN were used as the system model. The reasoning behind including MLR is that it is the simplest possible prediction model and is easy to interpret the results. Due to this we will use MLR as a baseline to compare every other method to. MLR is the linear combination of the weighted process variables, as described in the equation below:

$$\hat{\mathbf{y}} = \sum_{i=1}^{d} a_i \mathbf{x}^i \tag{2}$$

Where a_i ($i = 1, 2, \ldots, d$) denotes the ith model parameter and $\mathbf{x}^i$ is a column vector ($N \times 1$) of $\mathbf{X}$, which is all the historical values of a single feature. Due to its simple summation structure, MLR is unable to handle well many of the non-linear systems found in the chemical

process industry without any more special considerations, for example using feature engineering to create a PLS model. For this reason, we have included two additional methods used in soft sensing tasks: the eXtreme Gradient Boosting (XGBoost), which is an ensemble decision tree where each consecutive tree trains on the errors of the previous one (Chen and Guestrin, 2016), and the Support Vector Regression (SVR) method, which fits a hyperplane along the multidimensional data points to minimize the error between the plane and the samples (Smola and Schölkopf, 2004). As the detailed description of these methods is not the aim of this work, a comprehensive explanation of these algorithms will be excluded.

2.2.1. Comparison of ANN and KAN

Both ANNs and KANs are made up of layers containing neurons and edges that connect the neurons of different layers, as presented in Fig. 2.

In the case of ANN, the neurons contain a weighted linear combination of the inputs, where the weights ($\mathbf{w}$) of the inputs are defined on the edges between neurons. The combination is followed by an activation function (σ) that provides the output of the neuron. In general, the output of neuron n in layer l (y_l here) is defined as:

$$y_{n,l} = \sigma_{n,l}(\mathbf{w}_{n,l}\mathbf{x}_{n,l} + b_{n,l}) \tag{3}$$

where $\mathbf{w}_{n,l}$ is the input edge weight vector of layer l and neuron n with the size of $1 \times N_{l-1}$, where N_{l-1} is the number of neurons in the previous layer. $\mathbf{x}_{n,l}$ and $b_{n,l}$ are the input vector ($N_{l-1} \times 1$ size) and the bias scalar of the same neuron. Deep ANNs are the continuation of shallow ANNs by adding multiple hidden layers in series, improving the approximation capability of the model. However this can lead to significantly more parameters, increasing required computing capacity to train the network. There have been many modifications of ANNs for different tasks (e.g. LSTM, CNN) and KAN can be considered a branch of research in this area.

KAN is a recently developed method that is claimed to potentially be able to replace ANNs in many applications. The basis of the method is the Kolmogorov–Arnold theorem, which states that an arbitrary multivariate function can be approximated as the linear combination of univariate functions (Kolmogorov, 1961; Arnold, 2009). In the case of KAN the univariate functions are B-splines, parameterized by the grid size and the location of the knots. These splines are located on the edges connecting the neurons, each edge having its distinct spline. This leads to the most important difference between ANN and KAN. With ANNs the weights of the edges are trained with fixed activation functions placed inside the neurons. For KANs the activation functions are the edge splines that are trained, the neurons only contain a simple linear combination.

The reasoning behind the creation of KAN was that by combining splines (exceptional univariate estimation, major issues with curse of dimensionality) and ANNs (low impact of curse of dimensionality, weak univariate estimation capability), the advantageous parts of both estimators would remain while compensating for the limitations. In the case of KAN the authors of Liu et al. (2024) defined a layer as Φ matrix of 1D functions with n_{in} and n_{out} dimensional inputs and outputs (using the notation system from the original paper (Liu et al., 2024)):

$$\Phi = \{\phi_{p,q}\}, \quad p = 1, 2, \dots, n_{in} \quad q = 1, 2, \dots, n_{out} \tag{4}$$

where $\phi_{p,q} : \mathbb{R} \to \mathbb{R}$ are the functions that contain the parameters learned during the training. The $y_{n,l} \in \mathbb{R}$ output of the neuron n in layer l (using similar notation system as in Eq. (3)):

$$y_{n,l} = \sum_{i=1}^{N_{l-1}} \phi_{n,l,i}(x_{n,l,i}) \tag{5}$$

where $\phi_{n,l,i}$ and $x_{n,l,i} \in \mathbb{R}$ are the ith activation function and input value connected to neuron n in layer l and N_{l-1} is the number of neurons in layer $l-1$. Generalizing the notation to $\phi(x)$, the output of the activation function is:

$$\phi(x) = w_b b(x) + w_s spline(x) \tag{6}$$

where $b(x)$ is a sigmoid activation function:

$$b(x) = \frac{x}{(1 + exp(-x))} \tag{7}$$

and $spline(x)$ is the linear combination of B-splines ($B_i(x)$):

$$spline(x) = \sum_{i}^{K} c_i B_i(x) \tag{8}$$

and c_i, w_b and w_s are trainable parameters of the activation function (Liu et al., 2024). Splines of degree k are piecewise polynomial curves fitted on the available data. As such they can provide an accurate estimator for the univariate sections of the network, which are the edges connecting the neurons.

In this work we have used the *Pykan* Python library created by the authors of Liu et al. (2024). For the activation functions we have used splines of order 3, and for simplicity during the testing of multilayer KANs we have used hidden layers with equal number of neurons and included the learning rate and L1 regularization rate among the variables to be optimized.

The hyperparameters of the networks were optimized and the resulting models were compared with each other. Every model used the MSE metric as the loss function during the training to enable comparing their training runs and whether they overfit or underfit. For every network training task we have used the Adam optimizer (Kingma, 2014).

For the optimization of the hyperparameters we have used the Python library *Optuna* (Akiba et al., 2019). This library contains various methods for parameter sampling between the provided constraints, ranging from random or grid-based searches to different Bayesian techniques. In the current work the default sampling algorithm was used, which is the Tree-Structured Parzen Estimator (TPE). TPE is a Bayesian optimizer, which is a model-based derivative-free method. Due to its implementation, TPE does not explore the unknown regions of the parameter search space, thus it can get stuck in a local optima instead of finding the global optimum. To solve this we have run the optimization task multiple times with random starting points, and filtered out the global optima from the results. The optimization took on the following strategy: to prevent the possibility of a random seed in the train–test splitter influencing the accuracy of the model, the same *Optuna* suggested parameter set was used in 5 retraining cycles with the random seed splitter generating new random datasets. The resulting accuracies were then averaged to create a generalized model accuracy, to filter out the effect of random seeds. The optimizer runs 300 optimization trials with 100 warmup trials, where the warmup trials are where the sampler collects output samples before starting to optimize the hyperparameters. This ensures adequate exploration of the search space. The inclusion of an early stopping trial pruner algorithm was not necessary for this case study as the training of the networks was achievable is an acceptable time.

2.2.2. Shapley-based explainability analysis

To assess the explainability of the optimized and trained networks, we have applied the SHAP package in Python (Lundberg and Lee, 2017a). The contribution of each feature is quantified by estimating the Shapley values, which are based on game theory. In machine learning tasks the goal of using Shapley values is to decompose the prediction made by the neural network for a given sample and then assign a numeric value to each feature. Local explainer methods work by providing an explanation for $f(\mathbf{X}, \theta)$ output function (the model parameters are already identified so the notation can be simplified to $f(\mathbf{X})$) for a single input of $\mathbf{x}$ from $\mathbf{X}$. As the interpretation of the original model is complex, let g be a simpler *explanation model* and $\mathbf{x}'$ a simplified input generated by $\mathbf{x} = h_x(\mathbf{x}')$. Local explainers try to ensure

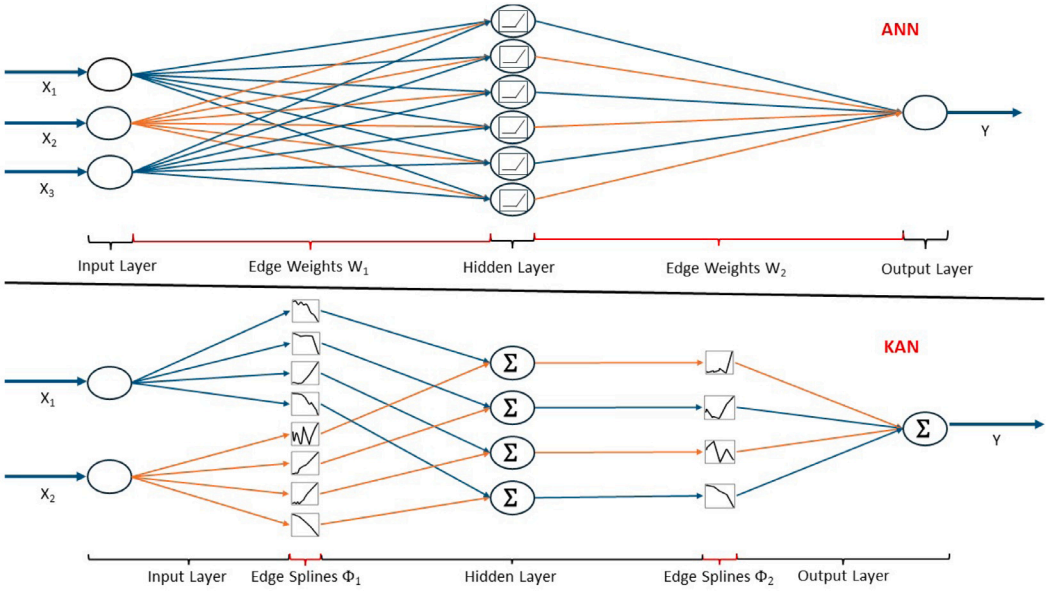

Fig. 2. Example of the structure of a general ANN and KAN (drawing made by the authors).

$g(\mathbf{z}') \approx f(h_x(\mathbf{z}'))$ when $\mathbf{z}' \approx \mathbf{x}'$ (where $\mathbf{z}'$ is a vector with $z_i' = 0$ if a feature is not in the model and $z_i' = 1$ if it is included) (Rozemberczki et al., 2022). The explanation model of the system is:

$$g(\mathbf{z}') = \psi_0 + \sum_{i=1}^{M} \psi_i z_i' \tag{9}$$

where $\psi_i \in \mathbb{R}$ is the effect attributed to the ith feature in $\mathbf{z}' \subset \{0,1\}^M$ and M is the number of simplified input features. The function describing attribution ψ_i is:

$$\psi_i(f, \mathbf{x}) = \sum \frac{|\mathbf{z}'|!(M - |\mathbf{z}'| - 1)!}{M!}[f_x(\mathbf{z}') - f_x(\mathbf{z}' \setminus i)] \tag{10}$$

where $f_x(\mathbf{z}') = f(h_x(\mathbf{z}'))$, $\mathbf{z}' \setminus i$ is the setting when $z_i' = 0$ and $|\mathbf{z}'|$ is the number of non-zero entries in $\mathbf{z}'$. The values of ψ_i are the Shapley values, the attributions of each of the features (Lundberg and Lee, 2017b).

SHAP includes the capability of displaying Shapley values of each feature for every sample on a diagram called a *beeswarm* plot. On these, the vertical size of a dot group for each feature shows its density, additionally the dots are colored based on the relative value of the feature. This means that it is possible to see whether the high or low value dots induce a positive or negative Shapley value, allowing the user to interpret and explain the effect of a feature using system knowledge.

3. Results

We have used two case studies to examine the capabilities of KAN. The first was a publicly available dataset from a steam turbine where the task is the prediction of the electric power. The second case study was the focal point of our work, where the task was the prediction of the rarely measured concentration of a by-product in an industrial aniline synthesis reactor.

3.1. Case study: Turbine energy output estimation

As a trial case study to check the capabilities of KAN at predicting and the interpretability of its symbolic formulas, we have applied it to a publicly available dataset. This was done on the electrical power output data of steam turbine, provided by the authors of Tüfekci (2014), Kaya et al. (2012). It is a dataset of 4 input (labeled as *AT*, *V*, *AP*, *RH*) and 1 output (labeled as *PE*) variables and contains 9568 data rows.

Table 1
Network hyperparameters.

Hyperparameter	ANN	KAN
Neurons	50	3
Layers	1	1
Learning rate (10^{-3})	4.3	1.7
Regularization rate (10^{-3})	0.7	1

The dataset was used as is, no preprocessing or feature engineering steps were done. First we constructed a KAN model using all input variables then using only the two most representative (*AT* and *V*). This selection was based on the visible relationships between the inputs and the output seen in Fig. 3. Both variables present a relatively clean and slightly non-linear trend which could necessitate the use of neural networks, proving that KAN is an appropriate method for this case study. The accuracy of KAN was compared to an MLR and a shallow neural network.

The hyperparameters of the two networks (ANN and KAN) are presented in Table 1. The parameters were optimized by hand and were found to be sufficient for the highly accurate representation of the output variable. Additionally to test our hypothesis that a simpler KAN network is easier to interpret and explain, the optimization included the minimization of the number of neurons and layers. First the four variable MLR, ANN and KAN were compared, the results of which are shown in Fig. 4. Every used model is capable of returning a high accuracy prediction, with MLR, ANN and KAN achieving R^2 values of 0.921, 0.933 and 0.927.

Following that the four and two variable KAN predictors were compared against each other. The resulting predictions on the testing dataset can be seen in Fig. 5. Both models predict the output with high accuracy, the four and two variable KAN having R^2 values of 0.927 and 0.923 respectively.

The next task at this stage was to use the built-in function of *Pykan* to generate a symbolic expression. It was done by pruning the network first to remove the edges with close to 0 weights. Then It is done by fitting predefined functions to the edge splines and selecting the best fit for each, then synthesizing the resulting expression from the functions. It was done for both cases, but as it can seen in Eq. (11) to Eq. (18) for the 2 variable case, even the smaller input set with only 3 neurons does not result in a by-design human interpretable equation. This highlights the need to use a different, post-modeling method of

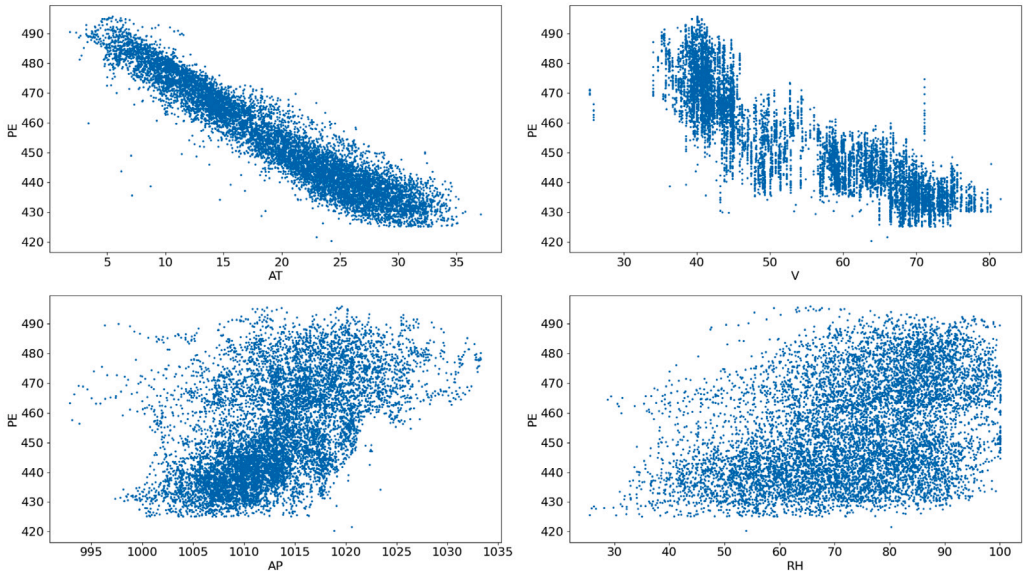

Fig. 3. Relationship between the inputs and output variable. AT and V have visibly clearer relationships with PE, hence they were used for the 2 variable model.

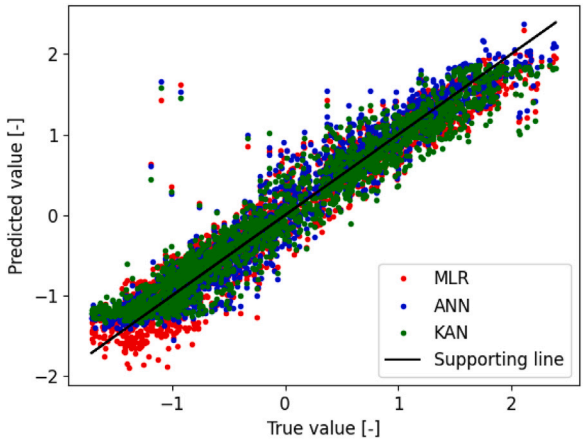

Fig. 4. Standard scaled predictions made by the tested models, plotted against the true output values.

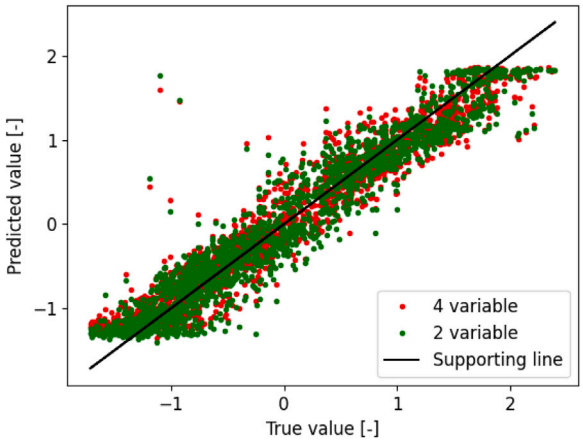

Fig. 5. Standard scaled predictions made by the KAN models using 4 then 2 input variables, plotted against the true output values. Both variable sets enable high accuracy predictions, thus interpretability becomes the next decision criteria when choosing which model to use.

extracting interpretability from the models, for this reason SHAP was used for all three potential nonlinear models in the case study of the aniline reactor.

$$\hat{y} = A + B + C + D \tag{11}$$

$$A = -0.6568\,\sin\left(0.6061\,x_1 - 0.5358\,\cos(1.502\,x_2 - 3.812) + 6.931\right) \tag{12}$$

$$B = -0.4584\,\tanh\left(0.4077\,(-x_2 - 0.2753)^2 - 0.3040\right) \tag{13}$$

$$C = -0.2423 \tag{14}$$

$$D = 1.262\,\exp\left(-5.181\,\left(D1 + D2 + D3\right)^2\right) \tag{15}$$

$$D1 = -0.04354\,x_2 \tag{16}$$

$$D2 = -0.8156 \tag{17}$$

$$D3 = \exp\left(-0.6925\,(-0.8915\,x_1 - 1)^2\right) \tag{18}$$

where x_1 and x_2 are AT and V respectively.

3.2. Case study: industrial aniline reactor

The dataset from a real-world industrial aniline synthesizing reactor was used as a case study to compare the applied models. It takes high-temperature mono-nitrobenzene (MNB), water suspended solid catalyst, and hydrogen, along with water for direct cooling (the reaction is highly exothermic). The system is a column reactor with multiple temperature sensor points along its length. The raw product is separated from the remaining components and further purified, while the side-stream liquid containing the catalyst is sent for catalyst recovery. The task of the soft sensor is to predict the concentration of a by-product component in the side-stream. This concentration is measured every 6 h, leading to large gaps between a change and possible corrective measures. Fig. 6 shows the structure of the reactor, here F1–F4 are inlet mass flow sensors for MNB, hydrogen, the catalyst and the cooling water. T1 is the inlet temperature, T2–T7 are the temperatures of certain trays in the column reactor. The sampling of the concentration happens after the sidestream is separated from the reactor outlet.

The available dataset contained 5463 samples, after the selection of the stable sections, it is reduced to 1784 samples. Of these, only 25% (446 points) had a matching quality value, showing the disparity between the availability of process and laboratory data. To avoid throwing away such a large amount of potentially useful data, we have

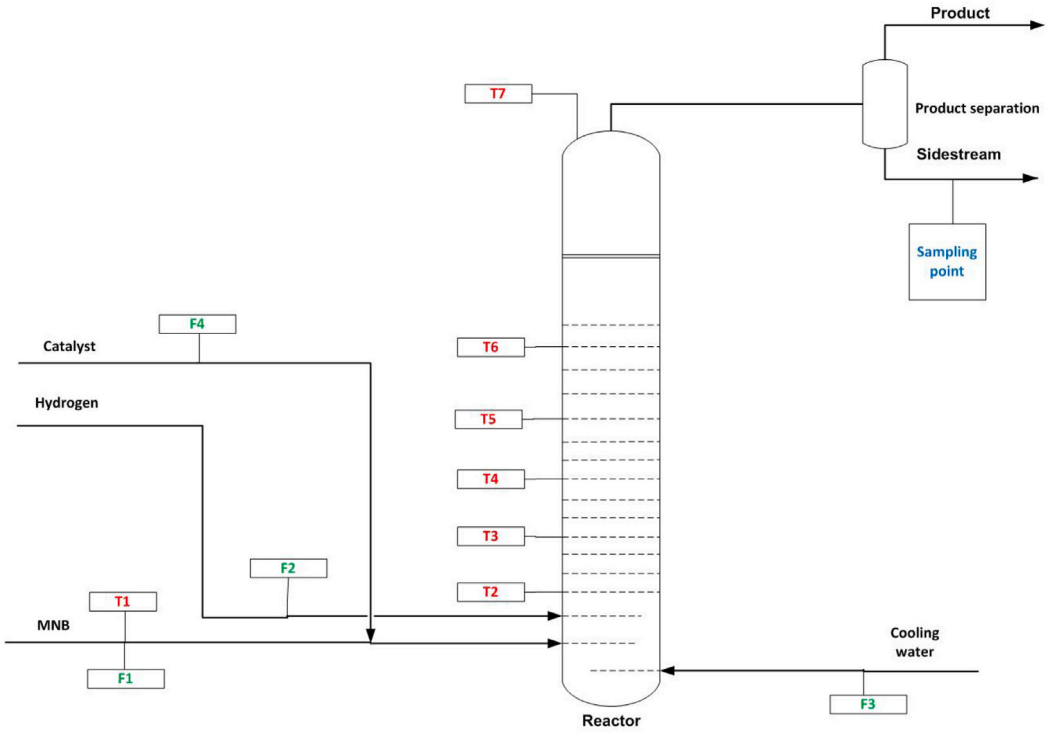

Fig. 6. Simplified process diagram of the aniline reactor used as case study (drawing made by the authors).

applied linear interpolation to fill in the rows between the real-life measurements, as it is a common and appropriate method to enlarge the dataset (Junninen et al., 2004). For the input variables, the number of missing samples was negligible, but in these cases we have applied the same linear interpolation as to the quality variable. Additionally, to account for the time constant of the system, we have matched up the process and quality variables. To do this we used Aspen Plus process simulator software to generate various water–MNB mixtures as a way to estimate densities and volume flows. The resulting volume flows were used to generate the average residence time of the reactor, which was in the 1.5–1.7 h range. Using this finding we have shifted up the quality variable column, matching every output with the input from 2 h ago.

3.2.1. Data preparation for reactor soft sensor development

As the example Fig. 7 shows, the preprocessed data contained extreme outliers that made the application of the 3σ rule (the valid samples are within the average $\pm 3\sigma$ range) not viable due to the average being distorted, and as such a preliminary manual outlier removal was done. As the focus of the work was the comparison of the KAN with other soft sensor models, we have considered this an acceptable solution. However for industrial implementation, a rigorous and automatic outlier detecting solution will be necessary. That task is outside of the scope of this work, hence will not be discussed. Without the outliers, Fig. 7(b) was achieved, and the same method was applied to all variables of the system.

The next step following the removal of outliers was that we have selected data sections where the value of MNB was close to stationary, as these defined operating points. We have selected a variety of sections from the whole dataset to best represent the system and its capabilities. The MNB flow defined the operating point, but the other controlled variables were changed during the operation of the reactor to keep the output stable. This means that the effect of the controlled variables were observable and modellable. Fig. 8 shows the data sections used for the feature selection and modeling task.

Afterwards the selected different stationary sections were further processed (standard scaled and split into train–test–validation sets) and then used for the modeling. As the states of a reactor depend on the inlet side of the system and the rate of reaction usually depends on concentration, the ratios of the inlet streams compared to the MNB inlet (namely F2:F1, F3:F1 and F4:F1) were used as modeling features along with the mass flow of MNB (F1). Additionally the T2 temperature was included in the feature set as it showed good correlation with the output.

Table 2
Ranges used for the hyperparameter optimization.

Hyperparameter	SNN	DNN	KAN
Neurons	[8, 512]	[8, 128]	[4, 64]
Layers	1	[2, 3]	[1, 3]
Learning rate (10^{-3})	[0.1, 100]	[0.1, 100]	[0.1, 100]
Regularization rate (10^{-3})	[0.1, 100]	[0.1, 100]	[0.1, 100]

3.2.2. Performance analysis of byproduct predictors

The ANN and KAN models were optimized by fitting them on training data with the selected hyperparameters, the test section was used for providing the objective function value to the optimizer. After the optimization the models were evaluated on the validation dataset, providing input and output data that is new to the sensor. All computations were performed on a workstation with an Intel Core i7-2600 CPU @ 3.40 GHz, 12 GB RAM and running Windows 11.

Due to the number of available samples, to prevent the possible overfitting of the deep ANN model, the number of possible hidden layers was in the range of [2, 3] and the number of possible neurons was [8, 128]. Along this to add a better possible representation capability to the shallow NN, its number of neurons was put in the [8, 512] range. For every network configuration the learning and regularization rate were constrained to the $[10^{-4}, 10^{-1}]$ range. For the KAN model the number of hidden layers was in the [1, 3] range, the number of neurons between 4 and 64 (as splines require more parameters than weights in an ANN layer). The optimization ranges are presented in Table 2.

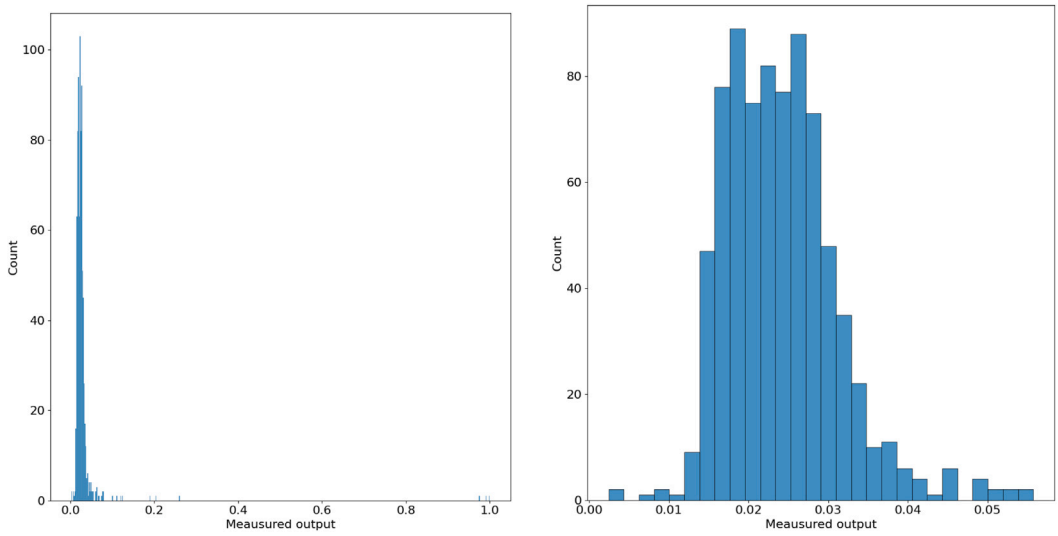

Fig. 7. Histogram of sample values before (a) and after (b) the removal of outliers.

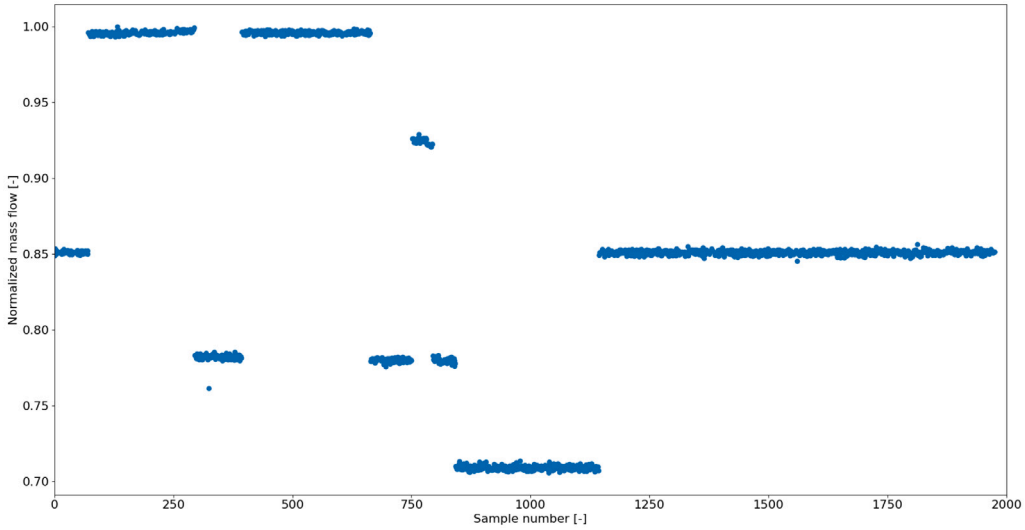

Fig. 8. Normalized mass flow sections used for modeling.

Table 3 shows the results of the hyperparameter optimization. One observation that could be of interest is that the learning and regularization rate of DNN and KAN are similar to each other, while for SNN both hyperparameters differ noticeably. This could be due to the multilayer structure both networks use, however to provide proof of that a fixed 1 layer KAN would need to be optimized and checked whether it is similar to the SNN or not. Additionally it would require multiple case studies to rule out coincidence, but as it is outside the scope of the current work it will not be done, only mentioned as a potentially interesting finding. The XGBoost model had 30 estimators with a maximum depth of 4, using the DART (Vinayak and Gilad-Bachrach, 2015) method and minimizing the squared prediction error. For the SVR we have used radial basis functions as kernel function.

Table 4 shows the results of the validation after the optimization and fitting steps were completed. To avoid the possibility of the initialization causing the weights to get stuck in a local optimum, each model fitting involved 100 runs (each iteration consisted of 250 epochs long training), the average results being presented in the table.

Table 3
Results of the hyperparameter optimization.

Hyperparameter	SNN	DNN	KAN
Neurons	444	100	15
Layers	1	2	2
Learning rate (10^{-3})	1.192	0.4050	0.899
Regularization rate (10^{-3})	0.1195	2.131	3.022

The MLR model showed major underperformance, hence it will not be involved in further analysis. The XGBoost and SVR algorithms managed to learn the trends in the dataset, however this trend prediction was paired with errors in inaccurate estimations. This could be due to XGBoost using regression trees that can learn patterns but only output a limited number of values, while SVR follows the data trends but the ϵ band used for its fitting can hinder regression accuracy. Both the boxplots and the table show that the 2 layer KAN is between SNN and DNN in terms of prediction accuracy and learning data trends with less parameters. However the relative difference is small between the

Table 4
Average results (± standard deviation for nonlinear models) of the model training where SNN: Shallow NN, DNN: Deep NN, SVR: Support Vector Regression, XGB: XGBoost.

	MLR	SNN	DNN	KAN	SVR	XGB
R^2	0.349	0.707 ± 0.041	0.721 ± 0.035	0.714 ± 0.039	0.676 ± 0.039	0.689 ± 0.046
RMSE	0.417	0.279 ± 0.023	0.269 ± 0.018	0.272 ± 0.021	0.701 ± 0.024	0.716 ± 0.025
MAE	0.297	0.208 ± 0.013	0.201 ± 0.012	0.206 ± 0.014	0.560 ± 0.039	0.714 ± 0.039
MAPE	18.16	10.33 ± 1.33	9.91 ± 0.98	10.16 ± 1.10	23.19 ± 1.24	24.30 ± 1.35
Params.	5	3139	10 831	2835	1035	480
Train time	0.001 s	43.34 s	43.24 s	17.71 s	0.116 s	0.069 s

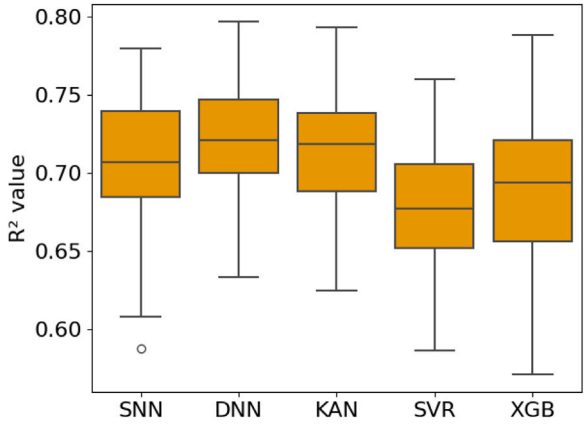

Fig. 9. Validation R^2 value boxplots for the investigated neural networks along with SVR and XGB.

different models. In terms of average training time, KAN is faster to train than the other networks, which could be due to the way the Python package is constructed or that fewer neurons are needed to backpropagate through. In comparison with the networks, the other models are faster to train by magnitudes, however those methods lack prediction accuracy in terms of trends or estimation precision.

Observing the quality predictions made by randomly trained networks (selected from above the medians of Fig. 9) compared to the real measurement values presented on Fig. 10 shows that every used network structure is capable of following the trends present in the dataset. However there are additional unobserved phenomena that influence the quality variable. One of these is that part of the catalyst is recycled later in the process, but its activity is not measured. Additionally, there was no information available on the laboratory measurement noise of the quality variable, thus the "flat" prediction sections between sample number 100–200 and 250–350 could be from model error or simply resistance to measurement noise.

As an additional test of model capabilities,we have modified the training data size while leaving the testing and validation data intact. This way we acquired a picture of how much information the different models could extract from available data, meaning how each could handle a task with a small available sample size. For this test we have removed 20%, 40% then 60% of the training data. The model training and evaluation was done using 100 runs, the same way as the previous test was done.

The results of this investigation are shown in Table 5, Table 6, Table 7. The results show a noticeable tendency of reduced accuracy across all metrics between 20% and 40% missing training data, affecting every used model. In this case the DNN model shows signs of stabilizing, in the RMSE, MAE and MAPE metrics the change between 40% and 60% is minimal.

3.2.3. Shapley-based product quality explanation

The previously described results show that every tested nonlinear model has a satisfactory level of accuracy, being able to track the general trends of the output. However in the modern process environment

Table 5
Average results (± standard deviation) of the SNN model training with different proportions of missing training data.

	R^2	RMSE	MAE	MAPE
0%	0.7066 ± 0.0414	0.2791 ± 0.0234	0.2078 ± 0.0128	10.33 ± 1.33
20%	0.7092 ± 0.0396	0.2729 ± 0.0215	0.2045 ± 0.0143	10.19 ± 1.13
40%	0.6796 ± 0.0409	0.2897 ± 0.0230	0.2167 ± 0.0158	10.91 ± 1.26
60%	0.6615 ± 0.0467	0.2968 ± 0.0221	0.2228 ± 0.0147	11.27 ± 1.23

Table 6
Average results (± standard deviation) of the DNN model training with different proportions of missing training data.

	R^2	RMSE	MAE	MAPE
0%	0.7206 ± 0.0353	0.2685 ± 0.0184	0.2009 ± 0.0123	9.91 ± 0.98
20%	0.7144 ± 0.0382	0.2713 ± 0.0209	0.2039 ± 0.0126	10.25 ± 1.08
40%	0.6887 ± 0.0444	0.2846 ± 0.0220	0.2138 ± 0.0138	10.72 ± 1.32
60%	0.6789 ± 0.0489	0.2863 ± 0.0230	0.2139 ± 0.0154	10.76 ± 1.35

Table 7
Average results (± standard deviation) of the KAN model training with different proportions of missing training data.

	R^2	RMSE	MAE	MAPE
0%	0.7141 ± 0.0385	0.2719 ± 0.0212	0.2060 ± 0.0141	10.16 ± 1.10
20%	0.7145 ± 0.0324	0.2733 ± 0.0197	0.2078 ± 0.0125	10.31 ± 1.03
40%	0.6877 ± 0.0376	0.2838 ± 0.0194	0.2158 ± 0.0126	10.70 ± 1.13
60%	0.6685 ± 0.0441	0.2907 ± 0.0204	0.2203 ± 0.0130	10.99 ± 1.05

it might be helpful to provide engineers and control room operators with the means to not just use blackbox models, but to interpret them too. This explainability could mean that a given output follows a trend that can be understood and explained by the user. It would lead to making decisions and changes to the operating points based on predictions that are reasonable according to real-life observations. As we have shown in our first case study, even few neurons and a simple dataset does not result in an equation that can be analyzed and interpreted. For this reason we have done Shapley analysis the 3 tested networks.

In the case of the three non-linear models, Fig. 11, Fig. 12 and Fig. 13 show the beeswarm plots of the optimized and trained networks using the complete available dataset (before being segmented into training, testing and validation data). Both SNN and DNN have similar shapes for F1, T2 and F3:F1 distributions, which can mean that the information extracted from these variables are similar. Besides that, interpreting the variables by their observed effect in the model and in reality, most of the features are used properly by the two models. Increasing F1 changes the amount of cooling water entering the system that can cool down the reactor before the exothermic reaction ramps up properly. A colder reactor can still convert the MNB and hydrogen feed due to the use of catalyst, however the catalyst can increase the reaction rate of the by-product synthesis at the expense of the aniline synthesis reaction. The same reasoning applies to the coolant water rate (F3:F1) and the catalyst feed rate (F4:F1) as the catalyst is fresh and recycled water suspended solid. The rationale behind the SHAP value assignment in the case of T2 is the reverse of what is observed in practice. Data analysis showed that higher temperature reactor was

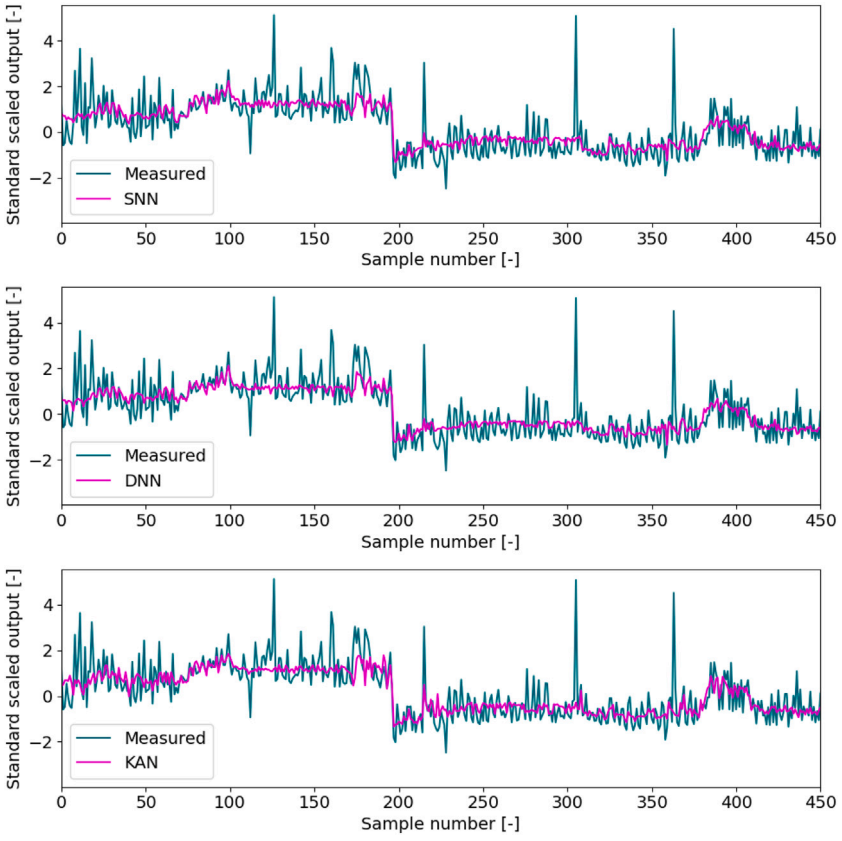

Fig. 10. Comparison of the real measurements and predicted outputs for SNN, DNN and KAN.

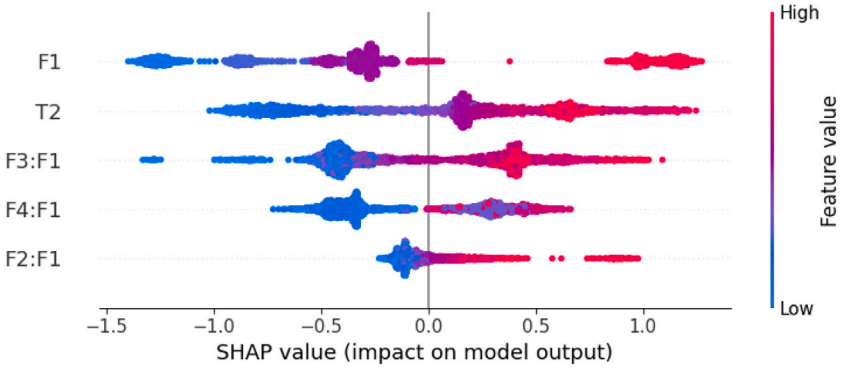

Fig. 11. Beeswarm plot of the SHAP values of the SNN.

found to decrease the by-product concentration, which could be due to hydrogen concentration decreasing faster and preventing unwanted reaction paths from taking place. Despite this, knowing this reverse value assignment can help understand how the networks operate. The value assignment of the hydrogen ratio (F2:F1) can be interpreted as the higher concentration H_2 can interact with the aniline product more readily, leading to more by-products.

The KAN beeswarm plot on the other hand is harder to interpret. On Fig. 13 it is harder to pinpoint clear trends in feature usage. For F1 and F4:F1 the SHAP values are distributed in a similar way as in Fig. 11 and Fig. 12, with points that have high relative feature value and negative SHAP value. These values can either be outliers or unobserved relationships extracted from the samples.

Analyzing the relative importance of each feature could provide additional information about the structure and inner workings of the networks. Fig. 14 shows that in SNN and DNN the features have a larger impact on the output prediction, while in KAN the features have a significantly lower absolute contribution. This was interpreted in two ways by the authors of this study: the larger contributions mean the networks could provide more resilient results as the less important noisy variables would not weight as much when generating the output. However this can be a drawback too if the sensors for the most important features are miscalibrated or fouled. In that case a mild measurement error can lead to magnified issues in prediction, leading to the operators changing the system in the wrong direction and producing large quantities of by-product. With KAN the assignment of less importance to every variable could be useful in utilizing every available variable, however it could also lead to giving weight of features that contain less information than we think, thus derailing the sensor estimations during application.

As the beeswarm plot for KAN shown on Fig. 13 provides less information, we have done sample specific attribution analysis for two outputs selected to include a high and low output scenario. The selected

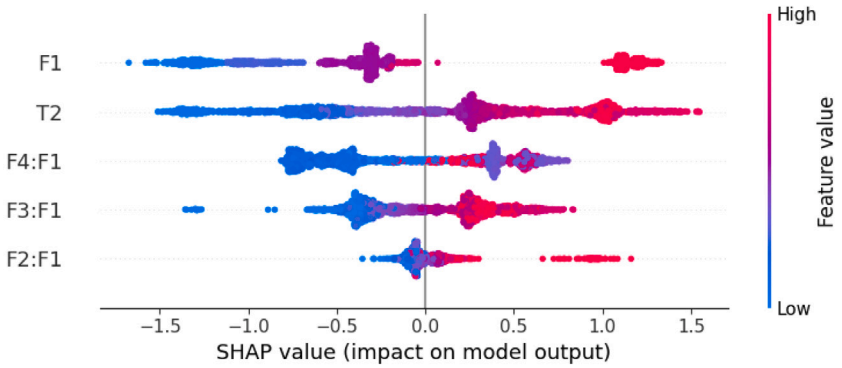

Fig. 12. Beeswarm plot of the SHAP values of the DNN.

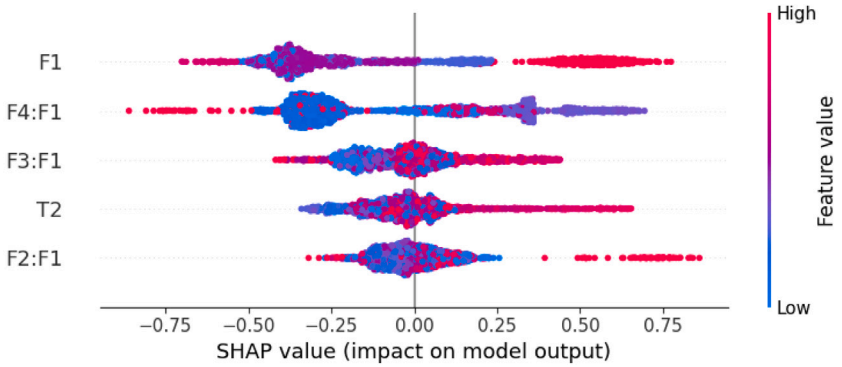

Fig. 13. Beeswarm plot of the SHAP values of the KAN.

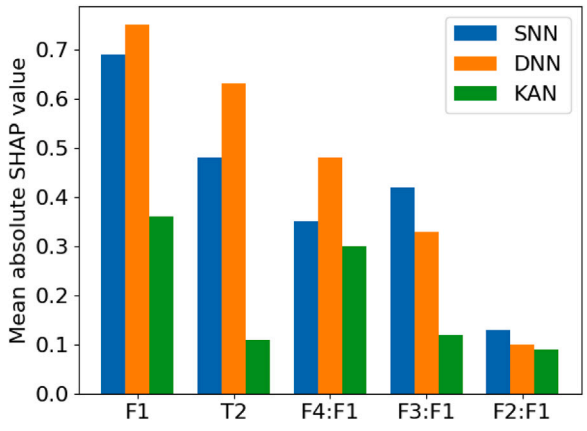

Fig. 14. Bar graph of the relative feature importance of each non-linear model.

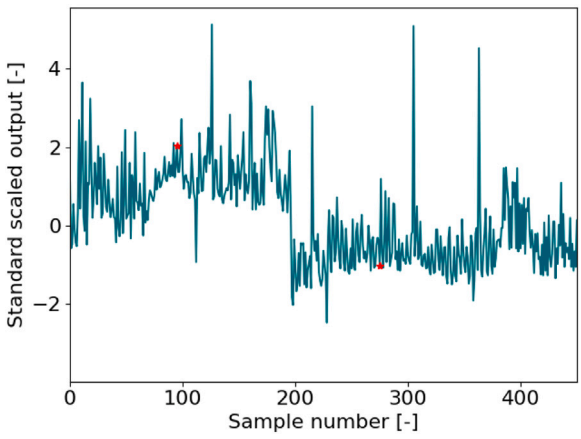

Fig. 15. Measured output with red dots indicating the samples examined for local SHAP attributions.

samples are from different sections of the real measured value dataset, as shown by red dots on Fig. 15.

The SHAP attributions for the two points are shown on Figs. 16 and 17. During the calculation of the individual attributions the expected value of the output is calculated and the attributions are added up to reach the final output of the predictor. On Fig. 16 it can be seen that the SHAP value assignment added the below average F1 value (shown as negative) a positive attribution, while on Fig. 17 there is a negative attribution. It can be understood in the context of the other variables. Low F1 value with above average ratios for F2, F3 and F4 cause the previously described effect of cooling down the reactor and allowing by-product synthesis and that these effects are building up on each other. On Fig. 17 a low F1 value is paired with below average F2 and F4 ratios while the F3 ratio is above the mean. Here the reverse of the previously described effect is visible. The low ratios drive the SHAP attribution of F1 down too, only the F3 ratio has positive attribution due to its high value. As described previously, the relationship between T2 and the byproduct seems reversed in the explanation, hence its interpretation is not clear. However it must be stated that these two points show two ideal situations in terms of attribution assignment. The global explanation on Fig. 13 show that the SHAP attributions are not that clear in general.

In summary it can be said that the Shapley value based explainability analysis found trends in the use of the variables by the model that resemble the effects seen in real life data. However every used feature interacts with the output in non-linear ways and also possibly influence the effect of the other variables. This means that while SHAP analysis

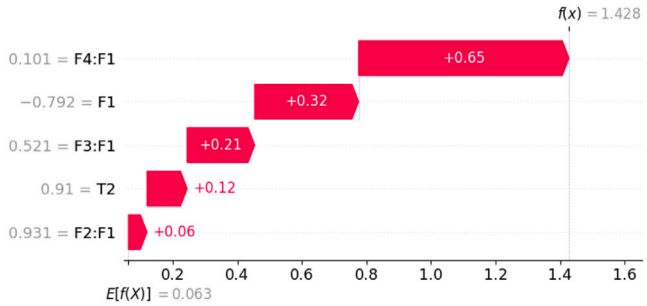

Fig. 16. SHAP attributions plot of the high output value scenario.

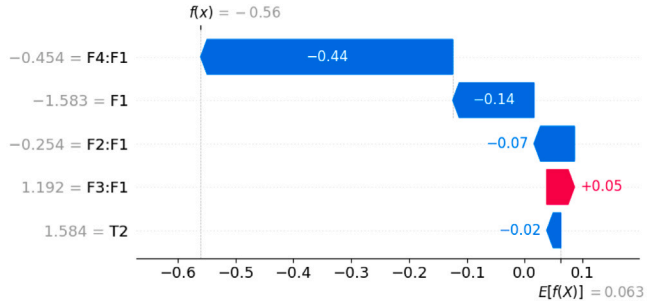

Fig. 17. SHAP attributions plot of the low output value scenario.

could be useful, it must be used carefully and with understanding of the process and potential feature interactions. Observing the beeswarm plots along with the local explanations it is clear that when compared to "traditional" neural networks, the KAN is less interpretable, a human operator or engineer would not be able to estimate the model output easily.

4. Conclusion

For this paper our intentions were to apply the Kolmogorov–Arnold Network as a soft sensor in two case studies, to test its symbolic regression capabilities to create interpretable models. Additionally, by using KAN as predictor in a complex production system, we were able to compare its accuracy to shallow and deep neural networks. Coupling it with Shapley-value based analysis, our aim was to provide an explanation of the output of the models by seeing if it is possible to identify trends in data use and match or interpret it with phenomena from practical engineering experiences. If human intuition matches how the model predicts the output, the user will trust the model more. An accurate model paired with interpretability is capable of supporting real industrial processes with data-driven solutions to minimize by-product formation, hence maximize product quality and optimize operational costs of the system.

Based on the conducted test the use of KAN as a model for a soft sensor is a viable alternative to various basic neural network structures as it has comparable accuracy after optimization. The trained model can be interpreted post-training if it is supported with expert knowledge of the process, hence predictions that are outliers can be overruled if needed in decision making situations.

KAN is a promising alternative to neural networks, however the method is rather recent and as such many aspects of its industrial applicability are not well explored. Among these is that process data is inherently noisy and the soft-sensor must be robust against it and provide a reliable baseline prediction even if it is less accurate. One of the main directions is the scalability of the KAN model for high-scale problems, where even the online monitoring, performance evaluation and retraining is also necessary. Also further industrial tests with varying, noisy and correlating data would expand the scope of the method from the small scale tasks and to potentially open new venues of application e.g. use as reinforcement learning agents. Expanding KAN to handle time series similarly to LSTM or convolutional networks or even spatio-temporal graph representations could show its strengths or application gaps, supplying engineers and researchers with a new tool for soft sensing. Additionally, supporting KAN predictions with methodology for the quantification of prediction uncertainty could strengthen its position as a future NN alternative. Our future intentions regarding KANs include advancing the previously mentioned areas of application, create various ensemble and hybrid models that enhance the applicability and performance of KAN.

Nomenclature

Abbreviations		Symbols	
ANN	Artificial Neural Network	a	Linear model parameter
DL	Deep Learning	$B(.)$	B-spline function
DNN	Deep Neural Network	D	Number of observations
KAN	Kolmogorov–Arnold Network	d	Number of input variables
kNN	k-Nearest Neighbor	$g(.)$	Local explanation function
MAE	Mean Absolute Error	$h(.)$	Simplified input generation function
MAPE	Mean Absolute Percentage Error	$\mathbf{h}$	Hyperparameter set
MLR	Multiple Linear Regression	k	Spline degree
MNB	Mono-Nitrobenzene	$\mathcal{L}(.)$	Objective function
MSE	Mean Square Error	M	Number of simplified input features
NN	Neural Network	N	Number of samples
PCA	Principal Component Analysis	$\mathbf{w}$	Edge weight vector of a layer
PI	Probability of Improvement	$\mathbf{x}^i$	ith model input variable
PLS	Partial Least Squares	$\mathbf{X}$	Input variable matrix
RMSE	Root Mean Square Error	$\mathbf{y}$	Selected output variable vector
SNN	Shallow Neural Network	$\hat{\mathbf{y}}$	Estimated output vector
TPE	Tree-Structured Parzen Est.	$\mathbf{Y}$	Output variable matrix
		$\phi(.)$	KAN activation function
		$\boldsymbol{\Phi}(.)$	Vector of KAN activation functions
		ψ	Attribution of a feature
		$\sigma(.)$	Activation function
		θ	Prediction model parameter set
		t	Date/time of observation
		τ	Time delay

CRediT authorship contribution statement

Balázs Fricz: Writing – original draft, Visualization, Methodology, Data curation, Conceptualization. **Gergely Horváth:** Writing – review & editing, Validation, Supervision, Conceptualization. **Alex Kummer:** Writing – review & editing, Visualization, Methodology, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This project was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences (B0/00439/25/6).

References

Abramov, A., Baranov, V., Filatova, A., Zasimova, M., 2024. Applying machine learning methods to forecast heat transfer characteristics in air-cooled single-row finned tube bundles under free convection conditions. Lobachevskii J. Math. 45 (10), 4858–4873.

Aha, D.W., Bankert, R.L., 1995. A comparative evaluation of sequential feature selection algorithms. In: Pre-Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics. PMLR, pp. 1–7.

Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M., 2019. Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2623–2631.

Andridge, R.R., Little, R.J., 2010. A review of hot deck imputation for survey non-response. Int. Stat. Rev. 78 (1), 40–64.

Ansar, T., Ashraf, W.M., 2025. Comparison of Kolmogorov–Arnold Networks and Multi-Layer Perceptron for modelling and optimisation analysis of energy systems. Energy AI 20, 100473.

Arnold, V.I., 2009. On functions of three variables. In: Collected Works: Representations of Functions, Celestial Mechanics and KAM Theory, 1957–1965. Springer, pp. 5–8.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W., 2010. A theory of learning from different domains. Mach. Learn. 79, 151–175.

Bradley, W., Kim, J., Kilwein, Z., Blakely, L., Eydenberg, M., Jalvin, J., Laird, C., Boukouvala, F., 2022. Perspectives on the integration between first-principles and data-driven modeling. Comput. Chem. Eng. 166, 107898.

Carvalho, D.V., Pereira, E.M., Cardoso, J.S., 2019. Machine learning interpretability: A survey on methods and metrics. Electronics 8 (8), 832.

Chaffart, D., Yuan, Y., 2025. Polynomial Neural Networks for improved AI transparency: An analysis of their inherent explainability (operational rationale) capabilities. Digit. Chem. Eng. 15, 100230.

Chen, T., Guestrin, C., 2016. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining. pp. 785–794.

Chen, Y., Wang, Y., Sui, Q., Yuan, X., Wang, K., Liu, C., 2024. Residual-aware deep attention graph convolutional network via unveiling data latent interactions for product quality prediction in industrial processes. Expert Syst. Appl. 245, 123078.

Doshi-Velez, F., Kim, B., 2017. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608.

Faubel, L., Woudsma, T., Methnani, L., Ghezeljhemeidan, A.G., Buelow, F., Schmid, K., van Driel, W.D., Kloepper, B., Theodorou, A., Nosratinia, M., et al., 2023. Towards an MLOps architecture for XAI in industrial applications. arXiv preprint arXiv:2309.12756.

Feng, L., Zhao, C., Li, Y., Zhou, M., Qiao, H., Fu, C., 2020. Multichannel diffusion graph convolutional network for the prediction of endpoint composition in the converter steelmaking process. IEEE Trans. Instrum. Meas. 70, 1–13.

Ferreira, J., Pedemonte, M., Torres, A.I., 2022. Development of a machine learning-based soft sensor for an oil refinery's distillation column. Comput. Chem. Eng. 161, 107756.

Fortuna, L., Graziani, S., Rizzo, A., Xibilia, M.G., et al., 2007. Soft Sensors for Monitoring and Control of Industrial Processes, vol. 22, Springer.

Foschi, J., Turolla, A., Antonelli, M., 2021. Soft sensor predictor of E. coli concentration based on conventional monitoring parameters for wastewater disinfection control. Water Res. 191, 116806.

Graziani, S., Xibilia, M.G., 2020. Deep learning for soft sensor design. Dev. Anal. Deep. Learn. Archit. 31–59.

Hu, Y., Xin, X., Yu, G., Deng, W., 2025. Deep insight: an efficient hybrid model for oil well production forecasting using spatio-temporal convolutional networks and Kolmogorov–Arnold networks. Sci. Rep. 15 (1), 8221.

Ji, T., Hou, Y., Zhang, D., 2024. A comprehensive survey on Kolmogorov Arnold networks (KAN). arXiv preprint arXiv:2407.11075.

Jiang, Y., Yin, S., Dong, J., Kaynak, O., 2020. A review on soft sensors for monitoring, control, and optimization of industrial processes. IEEE Sensors J. 21 (11), 12868–12881.

Junninen, H., Niska, H., Tuppurainen, K., Ruuskanen, J., Kolehmainen, M., 2004. Methods for imputation of missing values in air quality data sets. Atmos. Environ. 38 (18), 2895–2907.

Kaneko, H., Arakawa, M., Funatsu, K., 2009. Development of a new soft sensor method using independent component analysis and partial least squares. AIChE J. 55 (1), 87–98.

Kaya, H., Tüfekci, P., Gürgen, F.S., 2012. Local and global learning methods for predicting power of a combined gas & steam turbine. In: Proceedings of the International Conference on Emerging Trends in Computer and Electronics Engineering ICETCEE. pp. 13–18.

Kim, S., Okajima, R., Kano, M., Hasebe, S., 2013. Development of soft-sensor using locally weighted PLS with adaptive similarity measure. Chemometr. Intell. Lab. Syst. 124, 43–49.

Kingma, D.P., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Knottenbelt, W., Gao, Z., Wray, R., Zhang, W.Z., Liu, J., Crispin-Ortuzar, M., 2024. Coxkan: Kolmogorov-Arnold networks for interpretable, high-performance survival analysis. arXiv preprint arXiv:2409.04290.

Kolmogorov, A.N., 1961. On the Representation of Continuous Functions of Several Variables by Superpositions of Continuous Functions of a Smaller Number of Variables. American Mathematical Society.

Kotriwala, A., Klöpper, B., Dix, M., Gopalakrishnan, G., Ziobro, D., Potschka, A., 2021. XAI for operations in the process industry-applications, theses, and research directions. In: AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering. pp. 1–12.

Li, X., Xiong, H., Li, X., Wu, X., Zhang, X., Liu, J., Bian, J., Dou, D., 2022. Interpretable deep learning: Interpretation, interpretability, trustworthiness, and beyond. Knowl. Inf. Syst. 64 (12), 3197–3234.

Lin, B., Recke, B., Knudsen, J.K., Jørgensen, S.B., 2007. A systematic approach for soft sensor development. Comput. Chem. Eng. 31 (5–6), 419–425.

Lin, B., Recke, B., Schmidt, T.M., Knudsen, J.K., Jørgensen, S.B., 2009. Data-driven soft sensor design with multiple-rate sampled data: A comparative study. Ind. Eng. Chem. Res. 48 (11), 5379–5387.

Liu, Z., Ge, Z., Chen, G., Song, Z., 2018. Adaptive soft sensors for quality prediction under the framework of Bayesian network. Control Eng. Pract. 72, 19–28.

Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljačić, M., Hou, T.Y., Tegmark, M., 2024. Kan: Kolmogorov-Arnold networks. arXiv preprint arXiv:2404.19756.

Lundberg, S.M., Lee, S.-I., 2017a. A unified approach to interpreting model predictions. Adv. Neural Inf. Process. Syst. 30.

Lundberg, S.M., Lee, S.-I., 2017b. A unified approach to interpreting model predictions. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 30. Curran Associates, Inc., pp. 4765–4774.

Nazemzadeh, N., Malanca, A.A., Nielsen, R.F., Gernaey, K.V., Andersson, M.P., Mansouri, S.S., 2021. Integration of first-principle models and machine learning in a modeling framework: An application to flocculation. Chem. Eng. Sci. 245, 116864.

Rato, T.J., Reis, M.S., 2017. Multiresolution soft sensors: a new class of model structures for handling multiresolution data. Ind. Eng. Chem. Res. 56 (13), 3640–3654.

Rozemberczki, B., Watson, L., Bayer, P., Yang, H.-T., Kiss, O., Nilsson, S., Sarkar, R., 2022. The shapley value in machine learning. arXiv preprint arXiv:2202.05594.

Saravani, M.J., Noori, R., Jun, C., Kim, D., Bateni, S.M., Kianmehr, P., Woolway, R.I., 2025. Predicting chlorophyll-a concentrations in the world's largest lakes using Kolmogorov-Arnold networks. Environ. Sci. Technol. 59 (3), 1801–1810.

Shao, W., Tian, X., Wang, P., Deng, X., Chen, S., 2015. Online soft sensor design using local partial least squares models with adaptive process state partition. Chemometr. Intell. Lab. Syst. 144, 108–121.

Smola, A.J., Schölkopf, B., 2004. A tutorial on support vector regression. Stat. Comput. 14 (3), 199–222.

Somvanshi, S., Javed, S.A., Islam, M.M., Pandit, D., Das, S., 2024. A survey on Kolmogorov-Arnold network. arXiv preprint arXiv:2411.06078.

Song, F., Guo, Z., Mei, D., 2010. Feature selection using principal component analysis. In: 2010 International Conference on System Science, Engineering Design and Manufacturing Informatization, vol. 1, IEEE, pp. 27–30.

Souza, F.A., Araújo, R., Mendes, J., 2016. Review of soft sensor methods for regression applications. Chemometr. Intell. Lab. Syst. 152, 69–79.

Tüfekci, P., 2014. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. Int. J. Electr. Power Energy Syst. 60, 126–140.

Vinayak, R.K., Gilad-Bachrach, R., 2015. Dart: Dropouts meet multiple additive regression trees. In: Artificial Intelligence and Statistics. PMLR, pp. 489–497.

Wang, P., Yin, Y., Deng, X., Bo, Y., Shao, W., 2022. Semi-supervised echo state network with temporal–spatial graph regularization for dynamic soft sensor modeling of industrial processes. ISA Trans. 130, 306–315.

Wu, H., Han, Y., Jin, J., Geng, Z., 2021. Novel deep learning based on data fusion integrating correlation analysis for soft sensor modeling. Ind. Eng. Chem. Res. 60 (27), 10001–10010.

Yan, W., Xu, R., Wang, K., Di, T., Jiang, Z., 2020. Soft sensor modeling method based on semisupervised deep learning and its application to wastewater treatment plant. Ind. Eng. Chem. Res. 59 (10), 4589–4601.

Yuan, Y., Chaffart, D., Wu, T., Zhu, J., 2024. Transparency: The missing link to boosting AI transformations in chemical engineering. Engineering 39, 45–60.

Yuan, X., Huang, B., Wang, Y., Yang, C., Gui, W., 2018. Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE. IEEE Trans. Ind. Inform. 14 (7), 3235–3243.

Yuan, X., Li, L., Shardt, Y.A., Wang, Y., Yang, C., 2020. Deep learning with spatiotemporal attention-based LSTM for industrial soft sensor model development. IEEE Trans. Ind. Electron. 68 (5), 4404–4414.