frontiers in Chemical Engineering

ORIGINAL RESEARCH
published: 04 November 2021
doi: 10.3389/fceng.2021.778876

Increasing Superstructure Optimization Capacity Through Self-Learning Surrogate Models

Julia Granacher[1]*, *Ivan Daniel Kantor*[1,2] and *François Maréchal*[1]

[1]Industrial Process and Energy Systems Engineering, École Polytechnique Fédérale de Lausanne, Sion, Switzerland, [2]Institute of Sustainable Energy, University of Applied Sciences and Arts Western Switzerland, Sion, Switzerland

Simulation-based optimization models are widely applied to find optimal operating conditions of processes. Often, computational challenges arise from model complexity, making the generation of reliable design solutions difficult. We propose an algorithm for replacing non-linear process simulation models integrated in multi-level optimization of a process and energy system superstructure with surrogate models, applying an active learning strategy to continuously enrich the database on which the surrogate models are trained and evaluated. Surrogate models are generated and trained on an initial data set, each featuring the ability to quantify the uncertainty with which a prediction is made. Until a defined prediction quality is met, new data points are continuously labeled and added to the training set. They are selected from a pool of unlabeled data points based on the predicted uncertainty, ensuring a rapid improvement of surrogate quality. When applied in the optimization superstructure, the surrogates can only be used when the prediction quality for the given data point reaches a specified threshold, otherwise the original simulation model is called for evaluating the process performance and the newly obtained data points are used to improve the surrogates. The method is tested on three simulation models, ranging in size and complexity. The proposed approach yields mean squared errors of the test prediction below 2% for all cases. Applying the active learning approach leads to better predictions compared to random sampling for the same size of database. When integrated in the optimization framework, simpler surrogates are favored in over 60% of cases, while the more complex ones are enabled by using simulation results generated during optimization for improving the surrogates after the initial generation. Significant time savings are recorded when using complex process simulations, though the advantage gained for simpler processes is marginal. Overall, we show that the proposed method saves time and adds flexibility to complex superstructure optimization problems that involve optimizing process operating conditions. Computational time can be greatly reduced without penalizing result quality, while the continuous improvement of surrogates when simulation is used in the optimization leads to a natural refinement of the model.

Keywords: superstructure optimization, surrogate models, active learning, energy systems, process design, mathematical programming, artificial intelligence, machine learning

OPEN ACCESS

Edited by:
José María Ponce-Ortega, Michoacana University of San Nicolás de Hidalgo, Mexico

Reviewed by:
Linlin Liu, Dalian University of Technology, China
Esbeydi Villicaña, Universidad Michoacana de San Nicolás de Hidalgo, Mexico
Luis Germán Hernández-Pérez, Universidad Michoacana de San Nicolás de Hidalgo, Mexico

***Correspondence:**
Julia Granacher
julia.granacher@epfl.ch

Specialty section:
This article was submitted to Computational Methods in Chemical Engineering, a section of the journal Frontiers in Chemical Engineering

Received: *17 September 2021*
Accepted: *06 October 2021*
Published: *04 November 2021*

Citation:
Granacher J, Kantor ID and Maréchal F (2021) Increasing Superstructure Optimization Capacity Through Self-Learning Surrogate Models. Front. Chem. Eng. 3:778876. doi: 10.3389/fceng.2021.778876

1 INTRODUCTION

1.1 Motivation

Transitioning from fossil resources for supplying the energy and material requirements for society is one of the most pressing challenges we are currently facing. In addition to generation of renewable electricity and capture and utilization of carbon dioxide, conversion of woody biomass residues from industry holds great potential for supporting the energy transition, providing alternatives to store energy in the form of fuel and serve as a renewable resource for products conventionally produced from fossil carbon carriers. Ensuring efficient and optimal usage of valuable resources naturally leads to consideration of an ensemble of integrated options to generate good design solutions; systematic approaches like computer simulation, mathematical modeling and superstructure optimization must be applied to realize such solutions. These numerical models generally offer high levels of accuracy and precision in their predictive capabilities, but lead to large-scale, highly-complex, mixed-integer nonlinear optimization models and face computational challenges (Cozad et al., 2014). Efficient solution generation is hindered by high computational cost, noisy function evaluation of simulation-based optimization models as well as solving issues due to complexity and non-convexity of rigorous first-order processes (Fahmi and Cremaschi, 2012).

In a general optimization problem, an objective function is optimized with respect to decision variables, while a set of boundaries, inequality and equality constraints are implied in the problem formulation. For simulation-based optimization, widely applied in process design, the constraints are often not available in algebraic form, but are rather computed for any given decision variables *via* numerical integrators, lookup tables, or other constructs without an algebraic derivative. This leads to problems when applying standard optimization approaches that require derivative information. In the case that derivatives can be approximated or evaluated, some standard simulation software solvers are capable of optimizing without the algebraic model, for example Aspen Plus (Aspentech, 2019). However, the reliability of simulations often decreases with the complexity of the problem. Therefore, specialized algorithms are necessary, which are designed to recover from simulator convergence failures. Furthermore, researchers are often confronted with the high computational cost and noisy function evaluation inherent in simulation-based optimization models. With numerical simulations, it is natural that noisy function evaluations arise; however, they hinder the creation of accurate derivative estimations. This is addressed by creating derivative-free optimization algorithms to solve optimization problems when derivatives are unavailable, unreliable, or too expensive to evaluate. However, these models often struggle to find the best solution, especially for constrained cases (Rios and Sahinidis, 2013). Another approach to reduce complexity has been to apply special decomposition techniques (Douglas, 1988). Further issues that arise from attempting to solve simulation-based mathematical programming models are that the complexity and non-convexity of rigorous first-order process models makes them impossible to solve (Fahmi and Cremaschi, 2012). To overcome this issue in superstructure-optimization problems, separate flowsheets are used for different process configurations and resulting costs are compared. Thus, the optimization takes a somewhat heuristic approach, which might prevent researchers from finding the optimal process configuration (Fahmi and Cremaschi, 2012).

Applications in multi-objective optimization problems including superstructures have shown increasing need for efficient construction and use of surrogate models. The presence of many possible alternative configurations and operating conditions makes calculating the flowsheet model at each iteration impractical (Teske, 2014). In such cases, the original data can be used in computational experiments to generate missing data points such as if a set of experiments were performed. This set of data can then be used to fit surrogate models to describe the simulated system. Using surrogate models, the system simulation with new operating points can be performed in less computational time than with the original model, and model results can be generalized without encountering flowsheet convergence issues. In addition to advantages stemming from reduced computational time, surrogate models are known to provide a significant degree of flexibility and to be efficient in optimization applications (Bhaskar et al., 2001; Jones, 2001). Furthermore, surrogate models enable the inclusion of uncertainty in the analysis of a conversion system.

1.2 State of the Art in Surrogate Model Design for Process Engineering

Simulation-based optimization is a common approach in process engineering, and significant work has been performed in the field of surrogate model development. Particularly in the domains of molecular chemistry, supply chain management, residential energy systems, and chemical engineering, a large number of surrogate modeling applications are reported (Mirkouei and Haapala, 2014; Hansen et al., 2015; Bode et al., 2020). Furthermore, a variety of literature focuses on the general methodology for how to optimize surrogate models (Huang et al., 2006; Davis and Ierapetritou, 2009). Different types of approximation models exist. Artificial neural networks (Smith, 1993) represent one popular method to approach modeling, optimization, and control of chemical process systems. Their behavior is based on the imitation a brain's neurons. Another method widely applied is the Kriging-interpolation technique (Simpson et al., 1998). This method provides a statistical prediction of a function by minimizing its mean square error. Other approaches are simple polynomial regression models, response surfaces methodology, multivariate adaptive regression splines and radial basis function. A comprehensive overview about different modeling types is provided in Jin et al. (2001).

In process system engineering, simulation-based optimization is mostly addressed using Kriging or neural network modeling to build surrogate models (Cozad et al., 2014). Artificial neural

networks (ANNs) have especially been applied in chemical process modeling (Nascimento et al., 2000; Fahmi and Cremaschi, 2012; Teske, 2014; Tock and Maréchal, 2014). Their ability to fill gaps in the search grid caused by the absence of analytical solutions has been acknowledged (Nascimento et al., 2000), as well as their competence to reduce the computational time considerably when integrated in optimization frameworks, without penalizing model quality (Tock and Maréchal, 2014). Queipo et al. (2002) showed that ANNs can replace time-consuming numerical simulation of a heterogeneous and multi-phase petroleum reservoir used for the prediction of complex permeability and porosity distributions. Fernandes (2006) used ANNs to model Fischer-Tropsch synthesis and maximize product yield. Teske (2014) developed a global surrogate model of a rate-based fluidised bed methanation reactor model that transforms wood into SNG using–among others–ANNs. Fahmi and Cremaschi (2012) optimized the design of a biodiesel production plant by replacing all subsystems in a process flowsheet model with surrogate models based around ANNs and thereafter solving a defined mixed-integer nonlinear problem. Nuchitprasittichai and Cremaschi (2013) optimized a conventional amine-based carbon dioxide capture process to minimize the capture cost using ANNs. Tock and Maréchal (2014) developed parameterized modes of a carbon dioxide capture process for predicting optimized thermo-economic performance including investment cost, heat demands, and corresponding temperature levels for a range of flue gas flowrate and carbon dioxide concentration. The optimized sub-problem was integrated into a global problem formulation combining energy flow, energy integration, and economic models with a multi-objective optimization strategy as described by Gassner and Maréchal (2009). The approach was applied to study a natural gas combined cycle process with flue gas re-circulation.

Apart from ANNs, polynomial- and Kriging-based approaches are applied for surrogate modeling in process design. A methodology for the optimization of steady-state flowsheet simulators using empirical surrogate models based on polynomial and Kriging approaches has been presented by Palmer and Realff (2002). According to Palmer and Realff (2002), the developed meta-models only require a small set of solutions obtained from the simulation and still allows the optimization to proceed. Caballero and Grossmann (2008) developed an algorithm for the use of surrogate models in modular flowsheet optimization of constrained, nonlinear problems. In this particular application of optimization of modular process simulators, the derivatives are not available and some unit operations introduced noise and thus prevented the calculation of accurate derivatives. Noisy black box models of distillation columns and reactors were substituted by Kriging interpolation models. Henao and Maravelias (2011) presented a framework for surrogate-based superstructure optimization of an anime-based carbon dioxide capture system. Kriging interpolation and ANNs were applied for designing surrogate models integrated in the superstructure. Hasan et al. (2012) performed process modeling, simulation and Kriging-based optimization of adsorption-based carbon dioxide capture technologies. Quirante et al. (2015) presented a superstructure optimization approach for distillation columns that applied Kriging interpolation-based surrogate models. It was shown that the applied models qualify to accurately represent the original system with less than 5% error. More sampling points included in the model yielded better accuracy in reproducing the actual simulation results. However, with the increase in number of sampling points, the computational time to calibrate the model also increased consequently. Particular note should also be made of research that has recently been performed by the Institute for the Design of Advanced Energy Systems (IDAES) (Miller et al., 2018). The institute seeks to address the gaps of commercial simulation packages and general algebraic modeling languages by developing advanced process systems engineering capabilities to support the design and optimization of innovative processes. In this context, a framework for the generation of process models has been developed, containing interfaces for exporting, loading, and restoring modeling results. Furthermore, the interconnection of the modeling systems with higher-level tools, work-flows, and user-interfaces are included (Miller et al., 2018). A library of models for common unit operations has been developed, allowing for the simple representation of each unit. The applied data-driven machine language repository contains regression tools for the development of property models for the kinetics and thermodynamics of a system. The application of the developed tools range from coal-fired power plants to the optimization of power generation networks and next-generation power plants. Sikorski et al. (2016) applied parametrization of a biodiesel production plant model and showed the input-output relations between process parameters and heat loads. For the surrogate design, polynomial and high-dimensional model representation methods were used, both proven as valid tools for representing the original processes. Pedrozo et al. (2020) built multi-variable piecewise linear surrogate models based on commercial simulation software results and capital cost correlations by solving Generalized Disjunctive Programming problems. The surrogates were integrated in a Mixed Integer Linear Programming (MILP) problem formulated to determine the optimal design. Mountraki et al. (2020) introduced an iterative approach for the systematic evaluation of property process parameters from experimental data compatible with commercial software. The approach was applied for modeling an existing biorefinery. Reviews of surrogate-based optimization are provided by Forrester and Keane (2009) and McBride and Sundmacher (2019).

Alternatively to using static data sets for training surrogate models, algorithms can be extended to permit dynamic decisions regarding which data to add best at the current state of the model training and evaluation process. The concept of active learning is especially valuable when labeling data is computationally expensive, or where the purpose is to update the model using data that arrive continuously (Settles, 2012). In active learning, the model is initialized with a small sample of data from the design space and iteratively enlarged in domains where it is most needed. The technique enables efficient surrogate construction with limited data points required. This can be compared to Bayesian optimization, in which one would like to know the next best measurement, assuming the current evaluation is the last one. The next best measurement is added to the data set based

on the predictions of a machine learning model and an acquisition function that determines the potentially most promising measurement. Thus, the approach is closely linked to hyperparameter optimization, as the goal of both is to choose data and model parameters in the most efficient way (Lookman et al., 2019; Jablonka et al., 2020). Over recent years, active learning approaches aiming to reduce the amount of labeled data have gained interest in various domains. Sener and Savarese (2018) used an active learning approach for image classification using convolutional neural networks. Shen et al. (2017) apply active learning to reduce the data in their deep learning algorithms for named entity recognition. Jablonka et al. (2021) developed an active learning algorithm to predict the Pareto dominant materials for the design of carbon dioxide capture with multiple objectives. Their algorithm systematically improves the accuracy of the estimated Pareto front.

Despite interest in other domains, active learning approaches have not yet been applied in process systems engineering. Presented studies for surrogate model design, especially integrated in optimization frameworks, instead follow a static approach of training and evaluating models. Most presented studies focus on the surrogate model development for one specific process system, comparing performances of different surrogate models rather than addressing the development of a generic tool for surrogate model design for different processes. Lastly, we observe that the prediction uncertainty inherent in model predictions is not addressed, from which the integration in optimization frameworks could largely benefit, as it would allow for not only predicting process results, but also giving an estimation for the quality of a prediction.

2 METHODOLOGY

The objective of this research is to develop an approach for designing surrogate models of chemical process units integrated in process and energy system optimization frameworks with an active learning strategy. The design aims at assisting the efficient replacement of time-consuming simulations of chemical processes with more flexible designs, while avoiding convergence issues in the upper level optimization. For ensuring reliable predictions and optimization results, it is important to not only design surrogates, but also to provide a measure of their performance. Furthermore, since labeling data is computationally expensive, it is desired to reduce the number of labeled data points as much as possible, while the surrogates will continuously improve themselves while being used in the optimization framework. Thus, the main features of the approach explored here are flexibility for adapting to process models of different size and complexity, convenient and efficient application, and the ability to predict the process conditions but also yield the quality of the prediction that is made. An early stage version of the method was presented in Granacher et al. (2021), the extended version is presented hereafter.

2.1 Highlights of the Proposed Methodology

Our method is suggesting a generic algorithm for the efficient and reliable design for surrogate models of chemical processes that can be integrated in energy system optimization frameworks and that are able to improve themselves continuously. Our **A**ctive **L**earning **A**rtificial **I**ntelligence (ALAI) instance is used in two phases, namely the design and the application. In the design step, initial data are generated from the original process model and a set of surrogate models is designed, evaluated and stored in the ALAI instance. In the application step, the generated ALAI instance is included in the optimization framework and used for solution generation, given that the prediction quality is good enough. Otherwise, the original, simulation-based process model is called, and the obtained data are used to improve the ALAI instance.

2.2 Energy System Design and Optimization Strategy

The design of process and energy systems is a complex task that requires systematic modeling and optimization approaches for generating solutions and evaluating them on economic, social and environmental aspects. In our approach, the optimization problem divided in two parts, namely the upper and the lower level.

The lower level contains superstructure models that comprise the mass and energy conversion in the units, as well as the mass and energy integration in the system. The methodology for superstructure modeling and optimization is adapted from Gassner and Maréchal (2009).

For each unit u in the system, energy and mass flow models are built to describe the conversion in the unit regarding process streams, physical properties, mass and energy balances and to obtain the characteristics of the interfaces offered for integration with other units. The integration framework follows the formulation of Kantor et al. (2020), a summary of the formulation is described here. Depending on the type of unit, flowsheeting models are developed to derive the mass and energy balances of the system. Assuming a set of possible units $\mathbf{U}$ and a set of possible system states $\mathbf{T}$, binary decision variables $y_{\text{use}}(u)$ and $y_{\text{use}}\ (u,\ t)$ describe whether a unit is installed, and whether it is used in the respective period t. Continuous decision variables $f_{\text{mult}}(u)$ and $f_{\text{mult}}\ (u,\ t)$ describe the installed size of the unit and the level of usage at which it is operated in each period t. Continuous variables $f_{\text{mult}}(u)$ are constrained by parameterized upper and lower bounds: $F_u^{\text{min/max}}$. The binary decision variables $y_{\text{use}}(u)$ and $y_{\text{use}}\ (u,\ t)$ are constrained by the bound Y_u that determines whether a unit is considered for the generation of results. In the superstructure model, those variables are related by the set of **Eq. 1**.

$$\begin{aligned} F_u^{\min}\cdot y_{\text{use}}(u) &\le f_{\text{mult}}(u) \le F_u^{\max}\cdot y_{\text{use}}(u) \quad \forall u\in\mathbf{U} \\ F_u^{\min}\cdot y_{\text{use}}(u,t) &\le f_{\text{mult}}(u,t) \le F_u^{\max}\cdot y_{\text{use}}(u,t) \quad \forall u\in\mathbf{U}, t\in\mathbf{T} \\ Y_u &\ge y_{\text{use}}(u) \ge y_{\text{use}}(u,t) \quad \forall u\in\mathbf{U}, t\in\mathbf{T} \end{aligned} \tag{1}$$

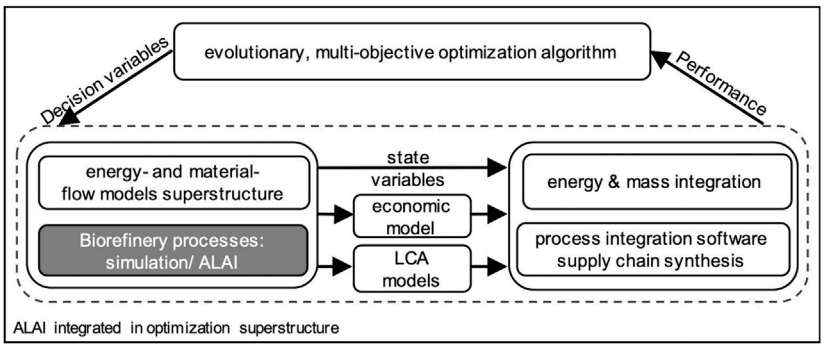

FIGURE 1 | Simplified methodology for process and energy system design , adapted from Kermani (2018).

In our approach, all units are available for heat integration, which allows for the exchange between the process and the utility systems to satisfy the mass and energy requirements of each unit. For the economic analysis, the operating and investment costs are calculated as a function of equipment size, using the cost functions available in Ulrich and Vasudevan (2003), Turton et al. (2018), for each unit. For the environmental assessment, the LCI (Life Cycle Inventory) Ecoinvent database (Wernet et al., 2016) is used to estimate the environmental impacts associated with waste streams and material use for the unit construction (Gerber et al., 2013). Pinch analysis (Maréchal and Kalitventzeff, 1998) and heat recovery for optimal utility selection is used to model the heat recovery and the integration of the utility system by introducing the heat cascade constraints explained in (Maréchal and Kalitventzeff, 1998).

The synthesis of configurations from this superstructure includes the manipulation of the decision variables by an optimizer. Decision variables of the lower level are either binary (e.g., unit installation $y_{\text{use}}(u)$) or continuous (e.g., unit size $f_{\text{mult}}(u)$). For generating a solution, the decision variables for a solution are determined by solving a MILP problem formulated in AMPL (Fourer et al., 2002), using the CPLEX (IBM, 2017) solver. For exploring the solution space of a given design problem and generating multiple solutions, the lower level is integrated in an upper level, in which an evolutionary, multi-objective optimization algorithm is used to solve a mixed-integer nonlinear programming (MINLP) problem by parameterizing the objective functions and sending the updated problems to the lower level (Gassner and Maréchal, 2009). Thereby, decision variables of the upper level are of nonlinear nature, such as the operating conditions of certain process units. For a problem communicated by the upper level, the lower level generates a solution and reports it back to the upper level (**Figure 1**).

2.3 Complexity Definition of Process Models

For evaluating the proposed method and identifying differences in model characteristics and performance for different processes, it is crucial to understand the complexity of a simulation model. Therefore, a ranking system is developed, quantifying the complexity of a simulation flowsheet based on multiple characteristics. Recent approaches determining model complexity focused on applying components from axiomatic design theory and information theory. Popovics and Monostori (2016) introduced an approach of defining simulation model complexity based on structural and software complexity measures. The structural complexity takes the number of modelled objects and the connections into account. For each model object, a set of attributes is considered. We follow a similar approach, where for each simulation model, the structural complexity C_{model} is defined by the number of equations N_{equ}, the number of variables N_{var}, the number of material and energy streams leaving and entering the system $N_{s,\text{in/out}}$, the number of reactions N_{react} as well as the overall unit complexity C_{units}. The overall unit complexity is computed as the sum over all unit complexities C_u, which are respectively calculated as the sum of unit input and output streams $N_{u,\text{in/out}}$. If the unit comprises mathematical equations for the evaluation of the performance, this is taken into account by Y_{flex} (**Eq. 2**).

$$\begin{aligned} C_{\text{model}} &= N_{\text{equ}} + N_{\text{var}} + N_{s,\text{in}} + N_{s,\text{out}} + N_{\text{react}} + C_{\text{units}} \\ C_{\text{units}} &= \sum_{u} C_u \\ C_u &= N_{u,\text{in}} + N_{u,\text{out}} + Y_{\text{flex}} \quad , Y_{\text{flex}} \in [0/1] \end{aligned} \tag{2}$$

2.4 Adaptive ALAI Design

The aim of designing a surrogate model is to replace the simulation-based units in the described lower level, for enhancing problem evaluation efficiency and robustness. In the design step of ALAI, the model must contain the relationship between the design variables (eg., operating conditions) and the system responses of complex thermo-chemical processes currently approximated by simulation software in the lower level (grey box in **Figure 1**). Thereby, the objective is to generate a reliable representation of the respective process with as few simulations as possible. Furthermore, the integration of the developed surrogate model in the energy system superstructure enables estimation of the prediction quality the surrogate model performs for unlabeled data, and respectively improve itself continuously by labeling the

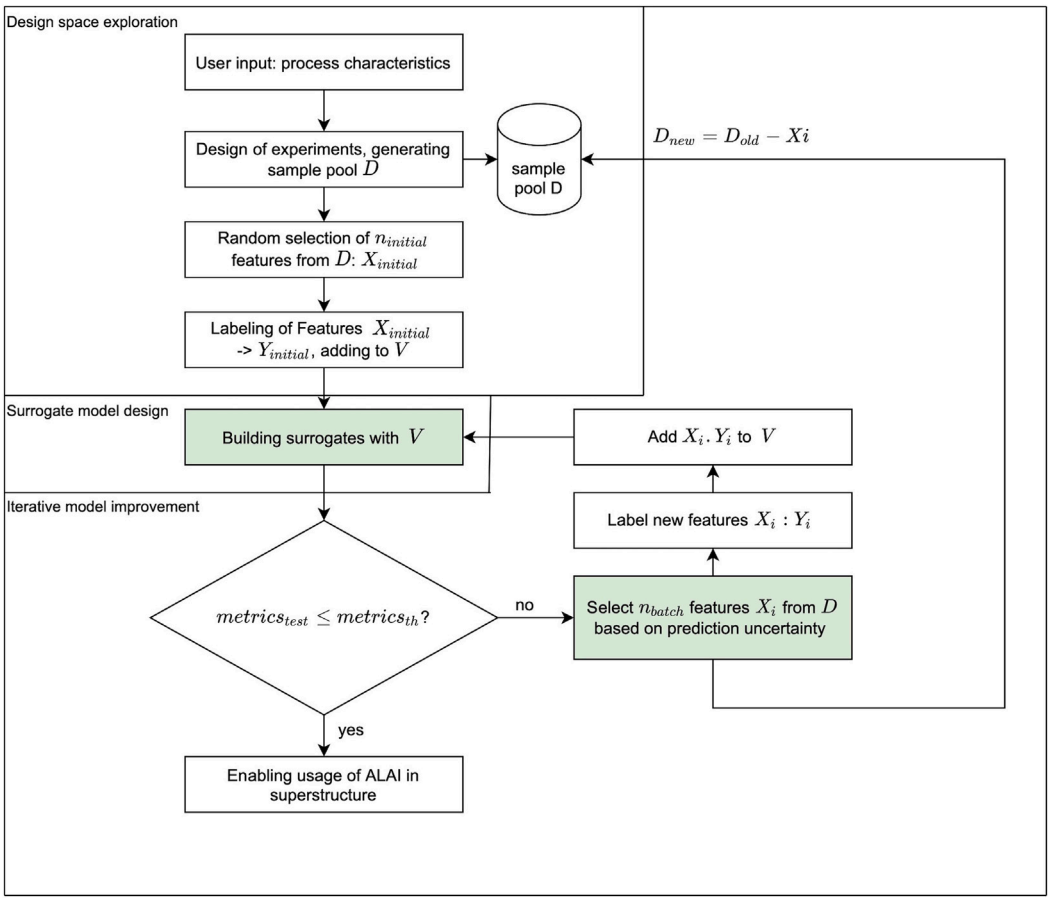

FIGURE 2 | Proposed algorithm for designing ALAI, green boxes are presented in more detail in **Figures 3**, **4**.

most uncertain feature samples and adding them to the ALAI training data set. For designing ALAI, the algorithm in **Figure 2** is followed.

2.4.1 Process Design Space Exploration

For better readability, in the following paragraphs "input" and "output" refer to characteristics of the original simulation models that are communicated to the energy system optimization model whereas "features" and "labels" are related to ALAI and the inherent surrogate models.

As a first step, the decision maker is asked to specify the process to be replaced by surrogate models, as well as d design variables relevant for the operation of the selected process unit. For the design variables, feasible ranges in which the process can operate must be defined. Design variables could include operating conditions of process units, but also efficiencies and compositions of relevant flows. Setting up the initial design of experiments, the design space D is created by drawing n_{total} samples for the design variables or inputs of the process to be replaced. Thus, D is of the dimension $d \times n_{\text{total}}$. For sampling the decision space, the quasi-random Latin-Hypercube sampling algorithm is applied, to systematically cover the entirety of the decision space (McKay et al., 1979). For a randomly-selected feature subset X of samples from D of the size $d \times n_{\text{initial}}$, the respective labels Y are created by calling the original simulation of the process and recording the outputs. Labels in this case are the outputs of the simulation that are needed by the energy system model formulation in the lower level for generating optimization results. They can contain stream temperatures, reaction heat loads, or any other process characteristics relevant in the mass and energy formulation of the unit. The ensemble of features X and labels Y used for training and evaluating surrogate models is stored in data set V. During the design and application of ALAI, it is continuously enriched.

2.4.2 Surrogate Model Design

For building the surrogate models in ALAI, the algorithm in **Figure 3** is followed. After parsing the data obtained from simulations to remove potential outliers and constant outputs, standard scaling is applied to transform features X and labels Y to distributions centered around zero and unity variance. For scaling, the mean and standard deviation of the training set are taken into account, avoiding data leakage during model evaluation. Furthermore, correlations between features are identified, as well as correlations between features and labels. Only features that are correlated to labels remain in the data set, and potential features correlated with others are removed. This

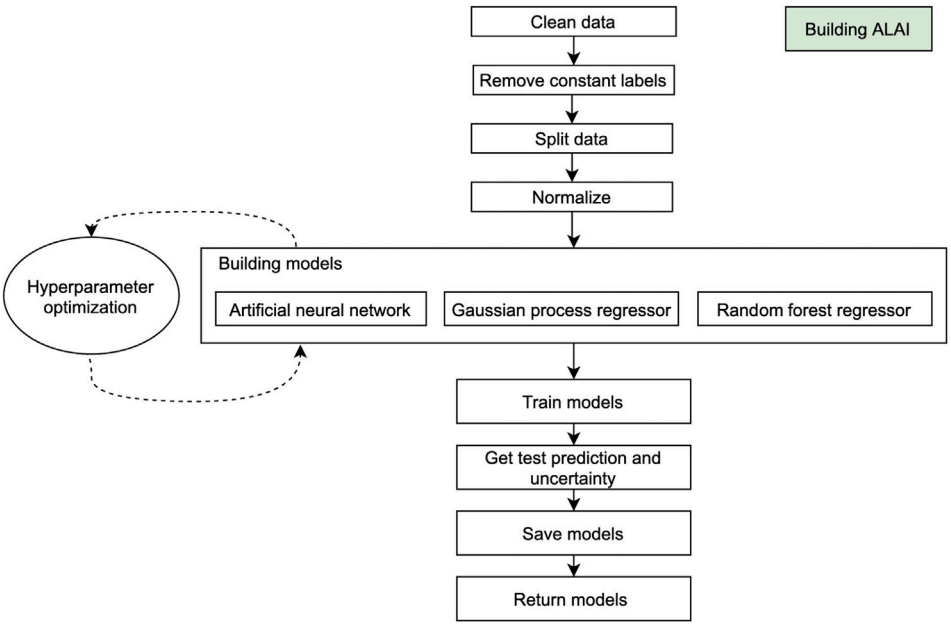

FIGURE 3 | Proposed algorithm for surrogate model design in ALAI.

latter step is not required if the decision maker can guarantee independent inputs. However, if dependency of certain model properties is unclear, this step can be performed prior to the building of ALAI on an initial data set obtained from simulation runs to inform decisions regarding input sampling for generating data from the simulation.

The selected subset is divided into a training and a testing set, with which ALAI builds and evaluates a set of surrogate models. Instead of building one fixed type of surrogate model, ALAI considers multiple types of deep learning and machine learning models. The reasoning for this is to enable ALAI to flexibly choose the best performing surrogate model based on the characteristic of the process unit and available data. A prerequisite of each surrogate model type to be accepted in ALAI is its ability to treat multi-label regression problems, and being capable of estimating uncertainty $\hat{\sigma}$ with which predictions $\hat{y}$ are made. By using the prediction uncertainty as a measurement of the quality of a prediction, ALAI can be continuously improved after the initial generation of the surrogate models.

2.4.2.1 Artificial Neural Networks

As deep learning (DL) components, ANNs are integrated in the ALAI superstructure. ANNs are computational models that are inspired by the nervous system of living beings. They can be interpreted as a set of interconnected processing units (neurons). ANNs hold many valuable features, such as their ability to adapt from experience, their learning competence or their capability to generalize (McCulloch and Pitts, 1943). Each neuron in a ANN gathers input signals from other units, assembles them and produces a response. Each of the input variables is affiliated with a weight w, which enables the quantification of their relevance with respect to the functionality of the neuron. The sum of the weighted inputs is saved in the linear aggregation l. If the result of the activation potential reaches the activation threshold Θ, the neuron produces an output signal. The activation function g limits the neuron output value h within a range of values. The output signal h can then serve as an input for other neurons in the network (da Silva et al., 2016) (**Eq. 3**).

$$\begin{aligned} l &= \sum_{i=1}^{1} w_i \cdot x_i - \Theta \\ h &= g(l) \end{aligned} \tag{3}$$

The structure of ANNs can be divided into three parts, namely the input layers, the hidden layers and the output layers. The input layer receives external signals and passes them on to the hidden layers. Usually, the inputs are normalized to improve the numerical precision of the mathematical operation. Hidden layers are responsible for extracting patterns associated with the analyzed system. In the output, final signals of the network are generated (da Silva et al., 2016). The training process of an ANN consists of tuning the synaptic weights and thresholds by using samples which represent the system behavior. After being trained on a subset of the data that describes the system, the ANN generalizes the behavior so that outputs can be predicted for any set of input parameters. Each complete usage of the training set to adjust the weights and thresholds is called an epoch. Optionally, hyper-parameter optimization is performed to improve the ANN's performance by optimizing the number of hidden layers, the number of neurons per layer and the activation function. Other hyper-parameters that can be optimized are the number of epochs, so the iterations used for fitting the model, as well as the batch size, which describes the number of samples considered per gradient update during training. For

approximating the prediction uncertainty for an unlabeled set of samples, we use dropout layers as suggested by Gal and Ghahramani (2016). According to Gal and Ghahramani (2016), an ANN with dropout applied before every weight layer is mathematically equivalent to a Bayesian approximation of the probabilistic Gaussian processes, which is an adequate tool to describe model uncertainty but at high computational cost. This problem can be overcome by transferring the concept of uncertainty to ANNs using dropout layers. In ANN design, dropout is typically applied to avoid over-fitting. When applying dropout, layer nodes in the ANN are randomly deactivated during training following a Bernoulli distribution. Consequently, a range is provided for each label by the ANN, allowing calculation of the mean and standard deviation for a prediction. For the complete theoretical method on how dropout is a valid representation of uncertainty, Gal and Ghahramani (2016) explains the approach in detail. In ALAI, an ANN including dropout layers is built with which–for each process condition evaluation request and each model output to be predicted–a range of predictions is obtained. This allows for computing the standard deviation and displaying it as a measure of prediction quality (Gal and Ghahramani, 2016). ANNs in ALAI are built in Keras, a deep learning API written in Python. Keras is running on top of the machine learning platform TensorFlow (Abadi et al., 2015). Hyperparameter optimization of the ANN is carried out with Scikit learn grid search using cross validation (Pedregosa et al., 2011).

2.4.2.2 Gaussian Process Regression

Gaussian process regression is a Bayesian approach especially known for its strong performance on small data sets. Unlike other supervised machine learning algorithms, the approach infers a probability distribution over all possible values instead of returning exact predictions for each. This enables the prediction of the next suitable point for evaluation by providing an uncertainty estimate, and for qualifying predictions. The Bayesian approach specifies a prior distribution, $p(w)$, on the parameter w and relocating probabilities based on evidence (i.e., observed data) using Bayes' Rule, where the posterior is calculated as the likelihood $p(y|X, w)$ times the prior $p(w)$ over the marginal likelihood $p(y|X)$ (Rasmussen and Williams, 2006):

$$p(w|y, X) = \frac{p(y|X, w)p(w)}{p(y|X)} \tag{4}$$

The posterior distribution incorporates information from the prior distribution and the data set. Predictive distributions at unseen points x^* are then calculated as

$$p(f^*|x^*, X, y) = \int_w p(f^*|x^*, w)p(w|X, y)dw, \tag{5}$$

where all possible predictions are weighted based on their calculated posterior distribution. The prior and likelihood are usually assumed to be Gaussian for integration to be tractable. Solving for the predictive distribution, a Gaussian distribution is obtained, from which the mean and uncertainty for a prediction point can be derived. Gaussian processes are non-parametric, which means that they are not limited by a functional form. Further explanation of the theory behind Gaussian processes can be found in Rasmussen and Williams (2006).

For implementing Gaussian process regression in the machine learning (ML) component of ALAI, we use GPY, a python library for Gaussian Process models, from the Sheffield machine learning group (GPy, 2012). We apply a Gaussian Process model for coregionalized multi-output regression, which becomes interesting when suspected that certain output dimensions might be correlated, as is likely the case for thermodynamic process design data. The model package for Latent Variable Multiple Output Gaussian Processes implemented by Dai et al. (2017) can be seen as an intrinsic model of coregionalization, if the coregionalization matrix is replaced by the kernel matrix. This replacement allows prediction of new outputs at test time, reduces the number of hyperparameters necessary to fit the covariance between the different conditions, and reduces overfitting when fewer data points are available for training. According to Dai et al. (2017), their approach outperforms related Gaussian coregrionalization processes significantly regarding computational efficiency. For building and optimizing our Gaussian process regression models we use RBF kernels and hyperparameter optimization using the limited memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm with 10 random restarts corresponding to the best maximum likelihood solution.

2.4.2.3 Random Forests

Random forests as proposed by Breiman (2001) are included as an alternative to Gaussian process regression and ANNs in ALAI. Random forest regression is a supervised learning algorithm combining predictions from multiple decision tree regressors to make a more accurate prediction compared to a single model (Biau and Scornet, 2016). Every decision tree regressor that the random forest contains can be seen as a piecewise constant approximation of the output. Apart from the training stage itself, the selection of hyperparameters is also decisive for the performance of the surrogate model, especially since the design must function with a large variety of process units. In our framework, we explore a range of hyperparameters in a grid search that use cross-validation Pedregosa et al. (2011), allowing to have a more robust estimation of the model performance. Explored hyperparameters include the number of trees, and the number of features used for building trees. For estimating the prediction uncertainty, the model structure of random forests is important. The random forest algorithm is a combination of a large number of estimators (decision trees) which, individually, may be subjected to variance but which, collectively, give the model the robustness it requires. Our approach of predicting uncertainty follows the method suggested by Meinshausen (2006), where estimators are used to model the conditional distribution function of the output. The inter-quantile range obtained from the set of estimators is used to measure distribution dispersion and thus quantify the uncertainty of the model.

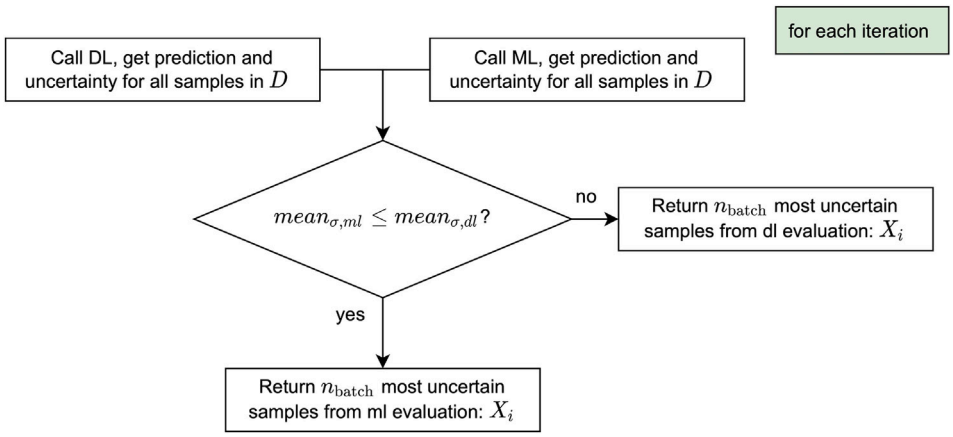

FIGURE 4 | Proposed algorithm for selecting n_{batch} when iteratively improving ALAI.

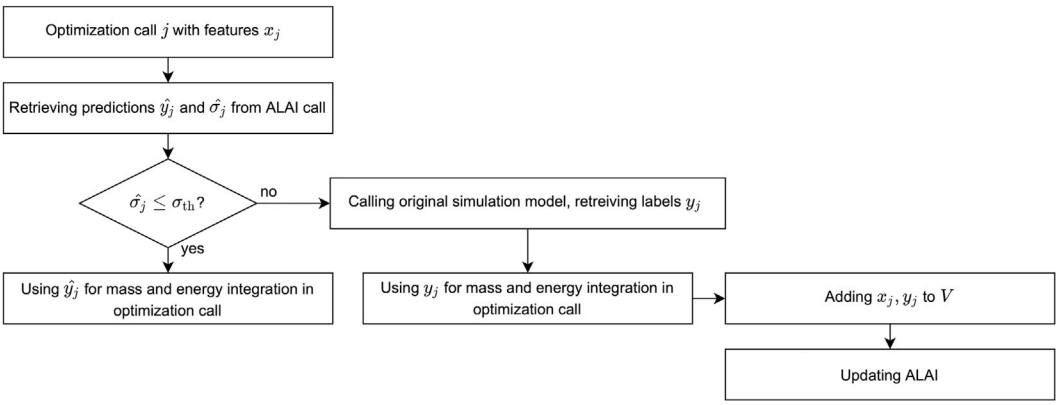

FIGURE 5 | Proposed algorithm for improving ALAI during application in optimization.

2.4.3 Iterative Model Improvement

After building the initial set of surrogate models, ALAI compares their individual performance on the test data and returns that which is best suited for making predictions for a given data structure. Alternative performance metrics could be used, but mean squared error is implemented as a default approach. If the desired test metric quality is satisfied for at least one of the included surrogate models, ALAI is accepted to be used in the optimization superstructure instead of the simulation software (**Figure 2**). However, if the performance is not satisfying, ALAI is improved by labelling more points in the data set V that is used for training and evaluation (**Figure 4**). For adding more data points to V, all surrogate models in ALAI are called to make predictions on the remaining samples in D. The surrogate model making the best predictions is determined by the mean uncertainty $mean_{\sigma}$ with which it makes predictions. A batch of samples with the highest uncertainty prediction for this model n_{batch} is selected for being labeled and added to V. After labeling, ALAI is updated with the new data. This procedure is repeated i times until the resulting test metrics satisfies the quality demands.

2.5 ALAI Application and Continuous Improvement

After the ALAI instance designed for a process unit has achieved sufficient quality for integration in the energy system superstructure, it can theoretically replace the call of the simulation software. When the energy system optimization model calls for a process unit simulation that has been replaced by ALAI, all surrogate models included in ALAI are called to predict the labels $\hat{y}_j$ for the features x_j given by the upper level optimization. Besides the resulting labels, the ALAI superstructure returns the predicted uncertainty $\hat{\sigma}_j$ for each surrogate model included in the superstructure, which is interpreted as the confidence which the surrogate model applies to the prediction. If the uncertainty of a prediction made by ALAI is lower than a specified threshold σ_{th}, the

prediction of ALAI with the highest confidence is used in the upper level optimization. Conversely, if ALAI returns a high uncertainty for a given prediction, the call is discarded and instead, the energy system optimization activates the original flowsheeting software to label the samples and carry out the optimization. In the latter case, in parallel to completing the optimization call with the generated labels y_j, x_j and y_j are added to data set V and ALAI is trained on the updated data (**Figure 5**).

3 APPLICATION

3.1 Process and Superstructure Description

The suggested approach of designing and applying ALAI instances is tested and validated on three process unit simulation models, namely a biomass to Fischer-Tropsch fuel process (FT), an ammonia reactor (AR) and co-electrolysis (CE). For all process simulation models, an ALAI instance is built and integrated in a multi-objective optimization framework, where selected operating conditions of the respective processes are optimized for economic performance using a genetic algorithm.

Ammonia is one of the most significant inorganic chemicals being industrially produced for societal needs, serving as the principle component for fertilizers in the food production system. Industrial ammonia is mainly produced in the Haber-Bosch process, where atmospheric nitrogen (N_2) is catalytically reacted with hydrogen (H_2) to produce ammonia (NH_3). The ammonia synthesis reactor simulation model is simplistic: it consists of 10 connected units, three system input and three system output streams. System input/output streams are defined as streams that connect from a process unit outside of the unit boundary. The simulation model features 60 variables and equations. Except three control units, all units have one or two input and output streams. The overall system includes one reaction taking place. Calculating the system complexity with the above described method leads to a score of 148 for the ammonia synthesis simulation model.

In co-electrolysis (CE), the concept leverages high-temperature solid oxide electrolysis cells (SOECs) for conversion of carbon dioxide (CO_2) and water (H_2O) into valuable chemicals. Our model is adapted from Zhang et al. (2017), taking into account ohmic, diffusion and activation losses, for a current density initially set to 0.3 A/cm^2 and an operating pressure of 1.5 bar. The operating voltage of the cell and stack, as well as the power consumption and co-electrolysis efficiency, are derived based on the process inputs and conditions. This process profits from high conversion and energy efficiency and offers opportunities for reducing CO_2 emissions; however, it is considered as an early-stage technology. Our co-electrolysis simulation model contains 31 units, 10 system input and 9 system output streams. Furthermore, each unit has between zero and six input and output streams. 120 variables are present, and thus 120 equations are evaluated. One overall reaction is taking place in the system. The system complexity calculation leads to a score of 324.

The Fischer-Tropsch reaction synthesis converts syngas with a given H_2 to carbon monoxide (CO) ratio into hydrocarbon liquids and waxy solids via a stepwise polymerization process. For producing Fischer-Tropsch fuels from lignocellulosic biomass, the biomass needs to be pretreated and gasified. Furthermore, the syngas needs to be cleaned and the ratio of H_2: CO needs to be adjusted to be suitable for FT reactions taking place. Our FT simulation model is adapted from Peduzzi (2015). It consists of 142 units represented in the simulation software, 23 system input and 26 system output streams. The simulation model requires 971 equations to be evaluated and features 971 variables and a total of 38 reactions. Each unit holds between 0 and 4 input streams and 0 and 6 output streams. The calculated system complexity is 2,391.

TABLE 1 | Operating conditions varied in case study.

AR	
Separation temperature (°C)	[270, 292]
Purge ratio (-)	[0.05, 0.2]
Reactor pressure (bar)	[270, 300]
Compressor efficiency (-)	[0.8, 0.95]
CE	
Reaction temperature (°C)	[950, 1050]
Reaction pressure (bar)	[1, 2]
Current density (A/m^2)	[0.15, 0.45]
Mole fraction (-) H_2O in	[0.003, 0.007]
FT	
Air inlet temperature (°C)	[180, 240]
Humidity a. drying (-)	[0.05, 0.35]
Torrefaction temperature (°C)	[210, 300]
Quench temperature (°C)	[700, 1000]

For all process simulation models, a set of input parameters that can be varied during operation and that influence the economic performance of the system are defined. For AR, four inputs are varied, including separation temperature, purge ratio, reactor pressure and compressor efficiency. For CE, four inputs are varied as well, namely the reaction temperature and pressure, the current density and the mole fraction of inlet water. For FT, five inputs are varied, modifying temperatures at various process steps and anticipated humidity fractions (see **Table 1**).

Building the ALAI instances of each described process crucially requires knowledge of which simulation outputs are to be predicted, hence, need to serve as labels. This information is retrieved from the integration of the respective process model with the utility superstructure in the optimization framework, which yields the information required for the optimization superstructure to construct mass and energy balances, e.g., thermodynamic states and extensive variables characterizing streams. With the identified inputs that should serve as features and the required outputs to serve as labels that need to be predicted, a first data set V for training and evaluating ALAI is generated by calling the process simulation model. The number of labels relevant for ALAI is further reduced by eliminating those that are not influenced by the varying inputs and remain constant throughout the design space exploration.

For the AR simulation process, seven outputs of the simulation are relevant for integration with the utility superstructure, of which two are found to be constant. The remaining five relevant outputs include the mass flowrate of ammonia out of the reactor,

TABLE 2 | Simulation model and energy system integration characteristics.

Process simulation models	AR	CE	FT
Number of equations	60	120	971
Number of unmeasured variables	60	120	971
Number of system input streams	3	10	23
Number of system output streams	3	9	26
Number of reactions	1	1	38
Overall complexity score of units	21 (10 units)	64 (31 units)	362 (142 units)
Overall system complexity	148	324	2,391
Integration with energy system optimization			
Inputs changed by energy system optimization	4	4	5
Number of relevant outputs	7	27	130
- constant outputs	2	5	67
- changing outputs	5	22	63

the volumetric flowrate in the reactor, the heat load, the input mass flowrate and the electricity requirements. For CE, 27 relevant outputs for the integration are identified, of which 22 vary with the inputs. They consist mainly of temperatures, heat loads, mass flowrates and electricity demands. For FT, an original set of 130 outputs from the simulations is needed for integration with the utility superstructure. From the 130 outputs, only 63 are affected by varying the selected inputs, while the rest remain constant. The 63 outputs include mainly heat loads and the corresponding temperatures, but also mass flowrates and electricity demands. For enabling the energy system optimization to perform heat recovery, it is crucial to receive the heating profile at a high temperature resolution, which results in a large number of outputs from the simulation model. **Table 2** summarizes the described characteristics of the simulation models.

3.2 ALAI Design and Application

For all three case studies, an initial design of experiments is performed generating $n_{\text{initial}} = 8000$ sampling points stored in D. A starting set of $n_{\text{initial}} = 10$ random sampling points is labeled and added to V with which ALAI is trained and evaluated. After preprocessing the data and training the ALAI components, the different surrogate models are evaluated based on the resulting test mean squared error (MSE_{test}), which is chosen as the evaluation metric for this case study. The threshold between the test predictions and the corresponding labels for accepting an ALAI instance to be used in the energy system optimization framework is set to $\text{MSE}_{\text{th}} = 0.02$ (2%), based on standardized labels. If, after the first generation of ALAI, the evaluation threshold is not reached, a batch of $n_{\text{batch}} = 10$ additional samples is added to the feature space of ALAI and labeled by calling the corresponding simulation models. For selecting which samples in D to add to V, all surrogate models of ALAI that were generated in the latest iteration are called to make predictions for the remaining data points in D. The surrogate model making the best overall prediction is determined based on the mean uncertainty that is reached for all predicted data points. The n_{batch} points with the highest prediction uncertainty are then selected based on the prediction of the identified best model. Once the ALAI instance has reached a sufficient evaluation metric, in this case a $\text{MSE}_{\text{test}} \leq 2\%$ for any surrogate model, it is added for being called in the optimization framework. When being called from the optimization framework, a prediction uncertainty smaller than $\sigma_{\text{th}} = 8\%$ is required for the surrogate model prediction from the ALAI instance to be used for generating optimization results. If this condition cannot be met, the flowsheet simulation model is called, and the generated data are used to improve ALAI.

4 RESULTS AND DISCUSSION

Overall, all three processes of interest can be represented with surrogate models after a certain number of iterations for generating the database V.

Since our selected value for n_{batch} was small, we let ALAI complete 10 iterations for each of the three processes despite the reached test metrics, which resulted in a size of V of 100. The simplest process (AR) was best represented with Gaussian regression (**Figure 6**), where a MSE_{test} below 2% if reached with only 30 data points; at 100 data points, MSE_{test} reaches 0.009% (**Table 3**). For the ANN, a MSE_{test} below 1% can be reached after 8 iterations, while the Random forest reaches an MSE_{test} of 2% with 100 data points, For CE, similar performance can be observed (**Figure 6**) as was found for AR. All surrogate models reach good performance for FT regarding MSE_{test} after 10 iterations. The Gaussian regression outperformed the other surrogate models, as it reached MSE_{test} below 1% with only 30 data points and reaches 0.07% with 100 data points. For the other surrogates, this accuracy cannot be reached in the datasize scope evaluated; however, the ANN and random forest methods both report decent performance as well with MSE_{test}s of 0.5 and 4%, respectively.

When observing the time required for ALAI to build the surrogate models, it should be noted that higher performance comes at the cost of training time. While generally, the training time for FT surrogates is longer than for the other processes due to the size and complexity, Gaussian regression is especially lengthy, showing the highest training time of the three surrogates. While the ANNs are trained within seconds or

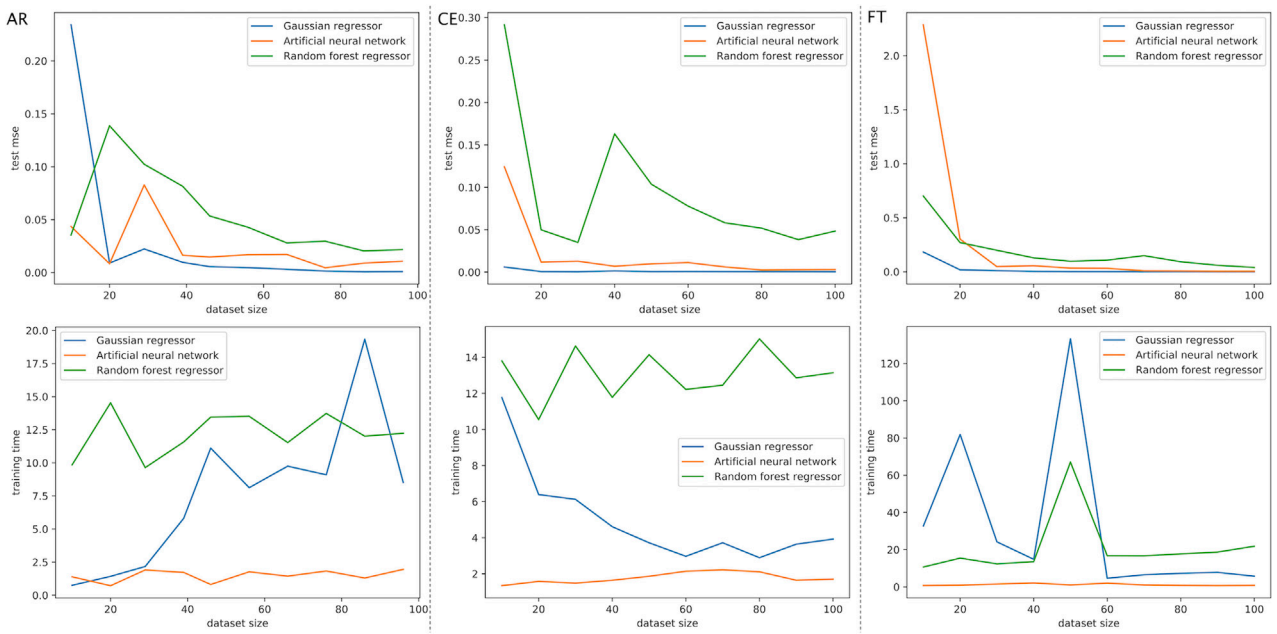

FIGURE 6 | Overall performance regarding test mean squared error and training time for analyzed case studies. Mean squared errors are plotted in absolute dimensions based on standardized data, whereas in the text, they are referred to in %.

TABLE 3 | ALAI MSE_{test} after 10 iterations, 100 data points used, absolute dimensions.

	AR	CE	FT
Gaussian regression	0.00093	0.00049	0.00073
ANN	0.01773	0.00305	0.00545
Random Forest	0.02187	0.04842	0.03844

fractions thereof, both the random forests and Gaussian regression require comparatively long training times. Here it may be noted that while the training time of the random forest increases with data set size, that of the Gaussian regression tends to decrease (**Figure 6**).

Overall, the results indicate that for all three case studies, a sufficient model performance regarding the defined metrics MSE_{test} can be achieved even with small data sets. Generally, higher process complexity yields consequently higher error in the first iterations of building ALAI; however, after 10 initial iterations, the reported testing errors approach the same magnitude for all processes. Gaussian regression seems to be an adequate surrogate model for the representation of a wide variety of process complexity, while ANNs perform better for more complex models. Random forests showed the weakest performance for the analyzed data sets, requiring long training times while predicting with relatively high errors compared to the other surrogate methods. This observation is likely related to the explored hyperparameter space, and adding more estimators or iterations during optimization could improve surrogate performance, at the cost of higher computational times.

4.1 Test Prediction and Uncertainty of Surrogate Models

Figure 7 shows the model evaluation progress regarding the test prediction accuracy and uncertainty for predicting the volumetric flowrate from the reactor in AR for different surrogate models in ALAI. **Figure 7A** shows the test evaluation for the random forest after five iterations. Generally, the test prediction does not provide a good estimation of the actual test labels, especially in the middle of the data set where fewer training data are available. The reported MSE_{test} for the prediction is 5.3%. Correlated to the poor prediction are the reported uncertainties: predictions further from test data logically yield higher reported uncertainty, indicated by whiskers of prediction. **Figure 7B** shows the same surrogate for the same process, but for a data set with 10 iterations. Predictions are generally more accurate–the MSE_{test} is reduced to 2.2%—but the same effect as previously observed can be seen, where regions with few data available result in worse predictions and higher predictions uncertainties than those with more data points available for training and testing. Prediction performance of the Gaussian regression for AR after 10 iterations is shown in **Figure 7C**. Here, the model performs well for all areas, resulting in low prediction uncertainty. For the ANN (**Figure 7D**), similar results emerge, although it is notable that the surrogate is underestimating the test labels especially in the upper domain for the given example. From this analysis, we acknowledge that ALAI is able to relate predictions with uncertainty measurements that indicate the

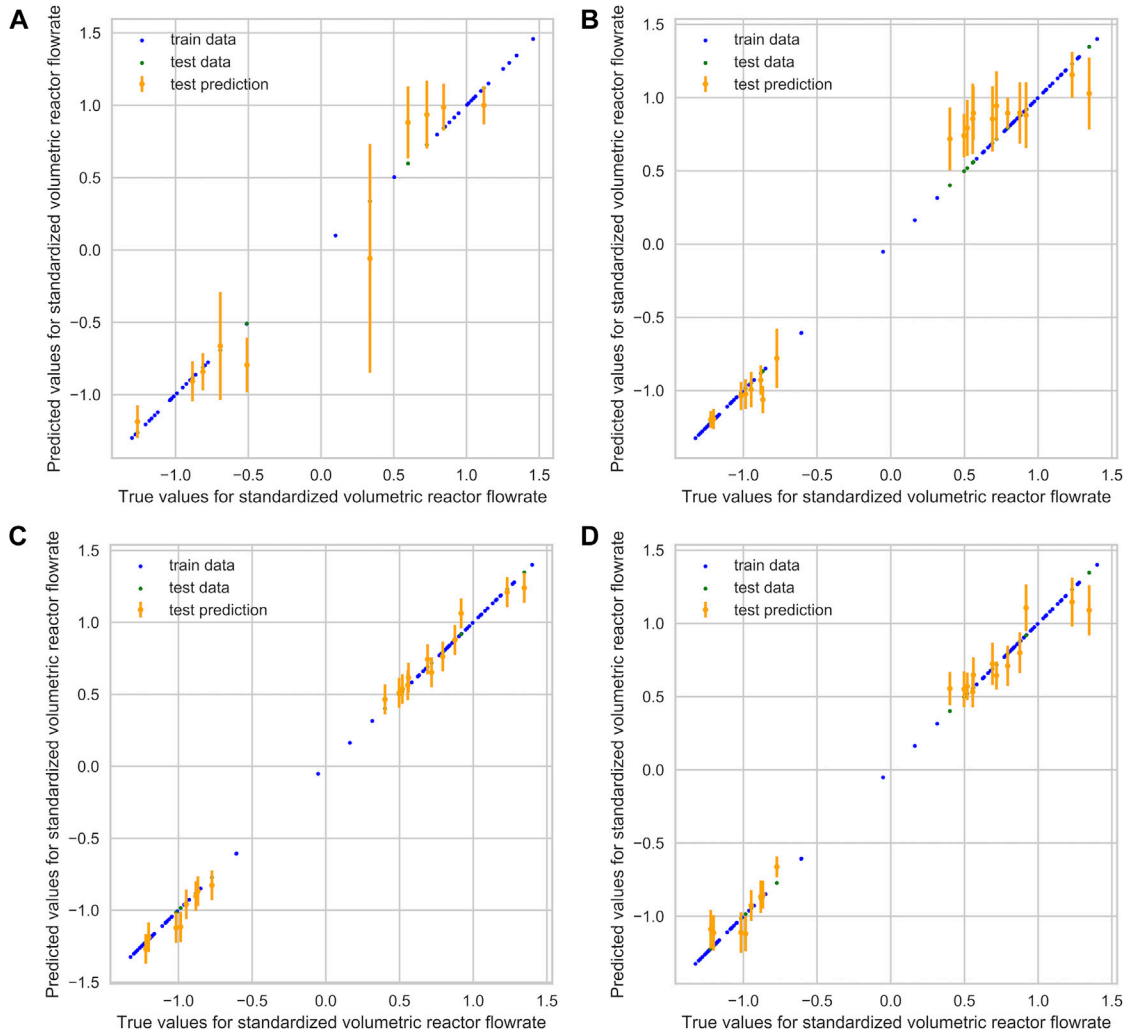

FIGURE 7 | Test predictions and uncertainty for AR. **(A)**: random forest with 50 datapoints, **(B)**: random forest with 100 datapoints, **(C)**: Gaussian regression with 100 datapoints, **(D)**: ANN with 100 datapoints.

actual quality of the prediction. Therefore, "bad" predictions are still useful, as long as they are identified as unsuitable.

4.2 Data Set Evolution During ALAI Instance Generation

In each iteration step of adaptive ALAI design, new samples are selected, labeled and added to the data set V based on the uncertainty prediction reported for the model with the best MSE_{test} in the previous iteration. This results in a model improvement with few samples to be labeled, as labels are only generated where they are most needed as ALAI makes poor predictions, which is–as shown previously–where the reported prediction uncertainty is high. The resulting order in which samples are labeled in each iteration is displayed for AR in **Figure 8**. The figure shows the features used in the model, reduced to two dimensions by applying the t-SNE visualization tool from Sklearn (Pedregosa et al., 2011).

It can be seen that the data points are added in a distributed order, each iteration contributing to improving the surrogate model performance. It should be noted that, as the samples for labeling are selected based on the reported uncertainties for the best performing models, it is evident that the best performing model improves more quickly than the others. It can be observed that typically the best performing model for a process stays the same throughout the iterations, which could indicate that after the initial iterations, only the best surrogate model could be kept in ALAI for further training, as the others do not show efficient improvement. However, in the presented study, we chose to keep all surrogate models in the evaluation for comparison and demonstration. In a later version of this method, earlier surrogate model selection after a certain amount of iterations could be considered to improve the computational speed.

We compare the presented active learning approach to a conventional labeling approach, where data points are randomly added to the training set. As an explanatory

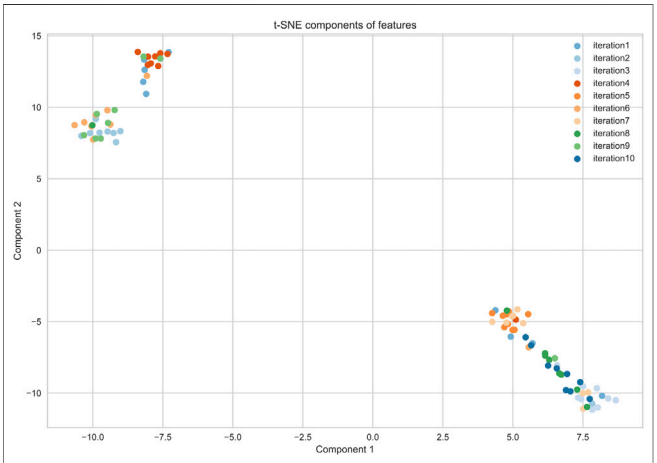

FIGURE 8 | Evolution of feature space during active learning, reduced to two dimensions.

example, we build an ALAI instance for FT with only a random forest surrogate model included, guaranteeing addition of data points in each iteration based on the reported prediction uncertainty of only this surrogate. Furthermore, we build another random forest for the same process, but instead of selecting new points for labeling based on prediction uncertainty, they are selected randomly. After 10 iteration, e.g., 100 data points in each data set, the MSE_{test} of the random forest created with the ALAI approach is 1.6% better than the one based on the randomly generated data set. **Figure 9** shows the test accuracy and uncertainty prediction for one example label of the process, comparing the performance of the random forest based on the random data set (**Figure 9A**) to the performance of the random forest generated with the ALAI database and its active learning approach (**Figure 9B**). Overall, the predictions made by the random forest in ALAI are more accurate, and thus the reported uncertainties are lower. Therefore, instead of generating large data sets of labeled model inputs when generating surrogates, a more efficient approach may initiate a small set and iteratively add new labeled points based on uncertainty predictions, so that time spent on labeling data is minimized, without penalizing the prediction quality.

4.3 Application of ALAI in the Optimization Framework

For integration with the optimization framework, a test metric threshold of $MSE_{th} = 2\%$ was initially defined. However, since we generated ALAI based on very small batch sizes, 10 iterations were evaluated for model performance, resulting in a data set 100 points. The demanded test prediction quality was achieved with a surrogate inherent in ALAI for all case studies. Thus, the ALAI instances of iteration 10 can be added to the optimization framework described in **Section 2.2** for replacing the original simulation models. The ALAI instance of the process of interest (henceforth referred as FT10, AS10, and CE10), is integrated with generic utilities and resources, such as electricity and heat supply, and a genetic algorithm is applied to find the Pareto front between capital expenditure (CAPEX) and operating expenditure (OPEX). For this, the size of the respective process to be integrated is set constant. The decision variables of the genetic algorithm are set to the varying operating conditions of the respective process, which also resemble the features of the ALAI instance. In each iteration, the genetic algorithm sets the decision variables and calls ALAI

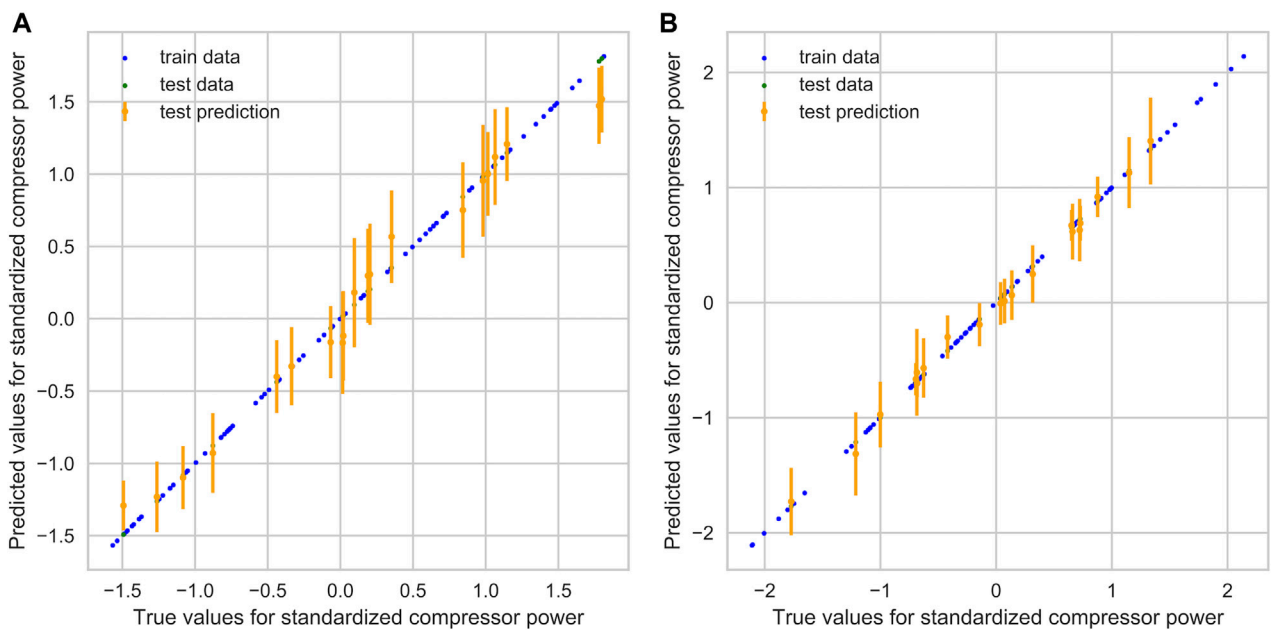

FIGURE 9 | Performance on test set with 100 labeled data points for FT. **(A)**: random selection of points to label, **(B)**: ALAI approach for selection of points to label.

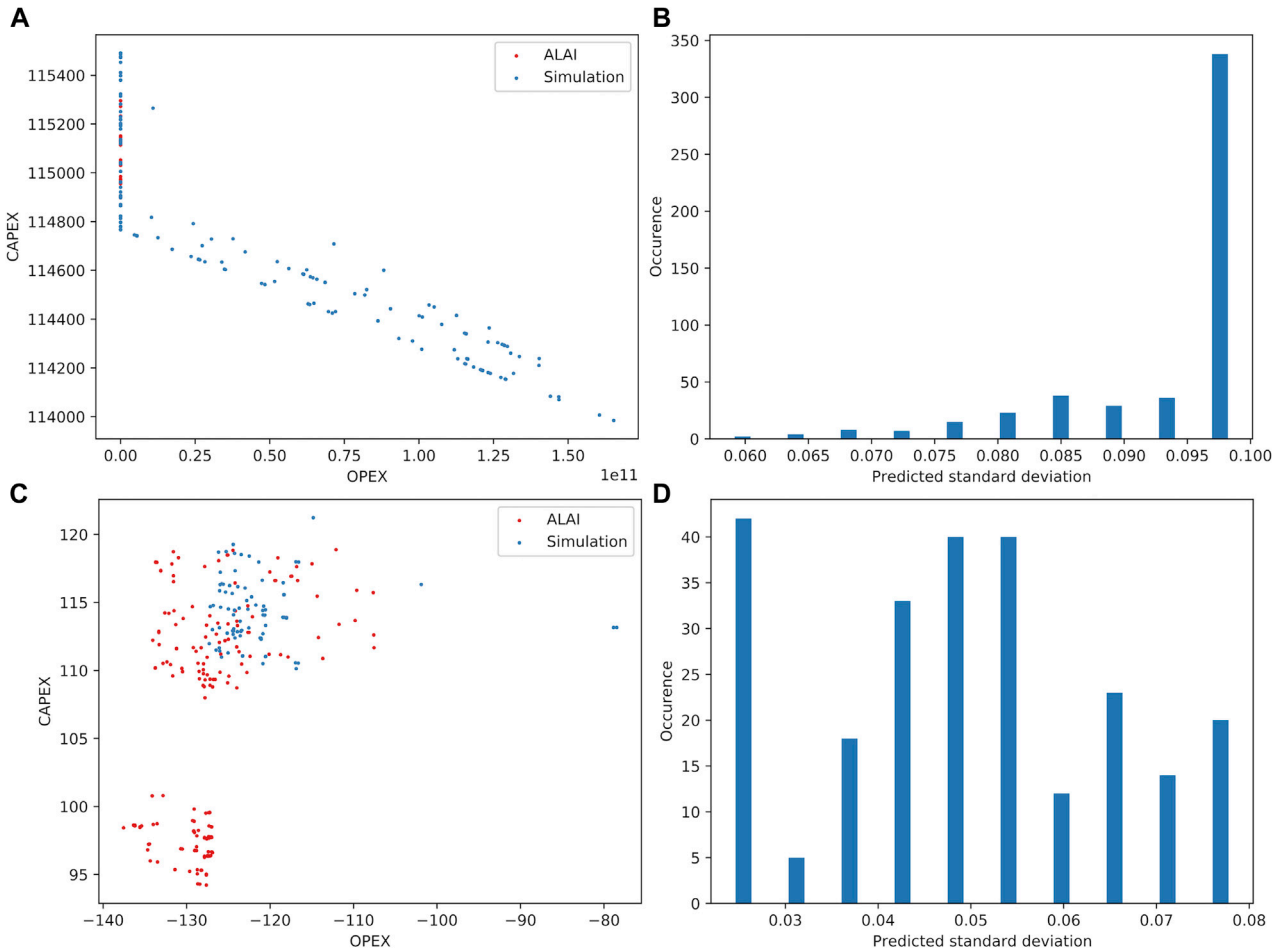

FIGURE 10 | Performance of ALAI instances for FT **(A,B)** and AR **(C,D)** integrated in the optimization framework with 500 optimization runs. **(A,C)**: Solution space obtained from optimization, showing operating condition evaluation method applied for each solution (ALAI or simulation). **(B,D)**: Distribution of predicted standard deviations from ALAI.

for predicting the resulting process conditions the optimizer needs to perform mass and energy integration between the process and the available utilities. For being accepted for usage in an iteration, the best predicted standard deviation of any surrogate model in ALAI needs to be below 8%. If, for a certain operating point, this requirement cannot be reached by ALAI, the original simulation software is called for generating the data needed for the optimization, and the results are recorded for later improvement of ALAI. When FT10 is integrated in the optimization framework using the genetic algorithm for conducting 500 optimization runs, the ALAI instance only performs well enough in 32 scenarios, requiring usage of the original simulation model, while for AR10, 356 of 500 ALAI calls are accepted (**Figure 10**). The different dimensions on the Pareto front come from the respective process size considered. For the Fischer-Tropsch process, the inlet biomass flowrate was set to 900 kg/h, while for the ammonia reactor, a fixed size of 15 kg/h is used.

Adding the labels generated by the simulation calls during the optimization result generation to V improves the performance of ALAI significantly. The improved FT10* is used in 598 out of 1,000 sample evaluations in the optimization framework, while AR10* is used in 813 out of 1,000 cases.

The results indicate that the frequency with which ALAI is called can be increased by using the optimization results for improvement of the surrogates, though considering that 8% uncertainty is acceptable. To explore the impact of this choice, a simple experiment was conducted on a sample where the prediction uncertainty is reported to 8%. A prediction with ALAI FT10 was selected, where the reported prediction uncertainty for the deep learning models (ANN) yielded 7.9% and the best uncertainty for the prediction made with a machine learning model in ALAI, Gaussian regression in this case, was at 9.1%.

The original FT simulation model was called for the respective sample to record the actual labels, and the results were compared to the recorded predictions of ALAI. Despite the fact that the prediction uncertainty was at, or even slightly over, the defined acceptance threshold in the optimization framework, the mean relative error between the 63 labels obtained from simulation and the prediction from ALAI are 0.7% for the DL prediction and 0.03% for the ML prediction. The maximum relative error

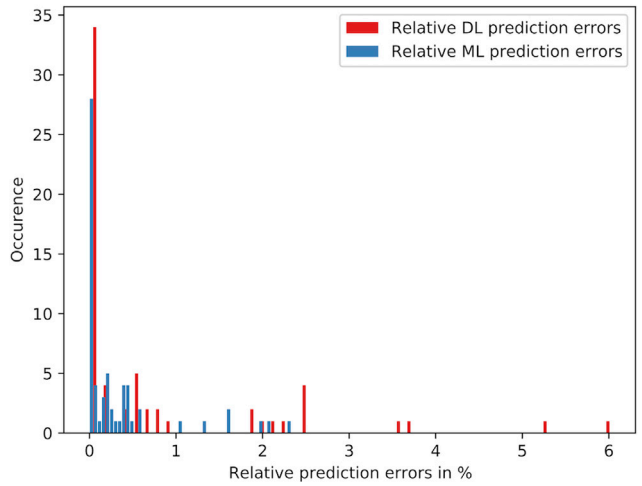

FIGURE 11 | Errors recorded between predictions and simulation results for a sample point with a predicted uncertainty of 7.9% for the DL instance and 9.1% for the ML instance of ALAI. A total of 63 outputs are predicted for FT, the occurrence indicates how often an error is recorded during the prediction.

recorded for the DL prediction accounts for 6% of the true value, for ML the error is at 2% (**Figure 11**). Thus, depending on the application of the optimization framework, the uncertainty threshold σ_{th} should be modified to achieve the acceptable error required.

Another important aspect for evaluating ALAI is computational speed. The time required for generating ALAI has been presented in detail, but the other crucial aspect is to evaluate the requirement for calling an ALAI instance, loading all surrogates, making the predictions and reporting which surrogate performs best for a given sample, including the process predictions and uncertainty. **Table 4** shows the comparison between the simulation times of the original models and the times reported for an ALAI call for all considered processes. The process conditions of the most complex model (FT) can be approximated with ALAI in less than 10% of the original simulation time. For CE, ALAI needs 67% of the simulation time, while for the simplistic AR model, there is no time reduction. A large part of the ALAI time is needed for making the ML predictions. Depending on the process, this accounts for 53–63% of the total reported time. The next largest time contribution is calling the ANN and making the DL prediction. Between 36 and 42% of the time is allocated to this stage. The rest of the time is used for loading databases needed for standardization, as well as the required software packages. Thus, depending on how often an ALAI call is accepted for being used in the optimization framework and the complexity of the original simulation model, a significant amount of time can be saved when used in the optimization framework. For the example discussed above, when FT10 is integrated for the first time, ALAI is called 500 times, out of which 32 calls are accepted. In the rest of the cases, the original simulation is called. This leads to 500 ALAI calls, and additional 468 simulation calls, which takes approximately 258 min. If ALAI had not been used in this case but only the simulation models, the time would have actually been less (252 min). However, since the simulation results of the integration were recorded and used to improve ALAI, the improved ALAI instance FT100* can be used in 598 optimization calls out of 1,000, which represents an acceptance rate of $r_{accepted}$ = 60%, resulting in an approximate time saving of 257 min or 50.8% compared to solely calling simulation. For CE10, the ALAI instance is accepted in 120 of the 500 cases, and for AR10, 356 ALAI instances are accepted. Due to the little time difference between the ALAI call and the simulation call, this leads to a time loss of 6 min for AR and 13 s for CE compared to allowing only simulation calls in the optimization framework. Therefore, the option of discarding the ability to call different surrogates from the optimization is added. When activated, only the model performing best regarding the latest MSE_{test} from an ALAI instance is called from the optimization. For a call of AR10, this leads to a reduction of the call time from 2.4 to 0.23 s and a time reduction compared to the simulation of 90%. In this case, where only one type of surrogate is called, ALAI would still be used in 99% of the optimization calls if the acceptance threshold was set to σ_{th} = 10%.

Besides discarding the call of individual surrogates, possibilities for improving ALAI performance during application are to improve the ALAI instances with the obtained simulation data during the initial application or to increase the uncertainty threshold. For example, when the acceptance rate is changed from 8 to 11%, 433 out of 500 ALAI calls (87%) are accepted for usage in the optimization for CE, which results in overall time savings of 390 s. When the simulation results recorded from the AR10 integration in the optimization framework are used to improve the ALAI instance, the MSE_{test} can be improved between 0.8 and 1.2% for the different surrogate models included. However, only 73% of the ALAI calls are accepted for usage with σ_{th} = 8%, which still leads to time penalties compared to pure simulation usage. In this case, it would make sense to discard surrogates as discussed before, or to augment the acceptance threshold.

4.4 Conclusion and Outlook

In this contribution, the methodology for replacing non-linear process simulation models integrated in a multi-level optimization framework of a process and energy system superstructure with surrogate models has been presented. An active learning approach is applied, where multiple surrogate models are trained and evaluated on data sets that are continuously increased based on the reported prediction uncertainty. It has been demonstrated that overall, our Active Learning Artificial Intelligence algorithm (ALAI) has the capability of generating reliable surrogate models that are able to predict the operating conditions of complex processes, even with small data sets. Different process model complexity showed varying performance among surrogate types. While Gaussian regression shows strong performance regarding test metrics for varying process complexity, the performance of ANNs increased with model complexity. Random forests generally performed better for simple process

TABLE 4 | Temporal performance of ALAI in optimization framework.

Case study	$t_{Simulation}$ (s)	t_{ALAI} (s)	Δ_t (%)	$n_{accepted,\ ALAI}$	$r_{accepted,\ ALAI,improved}$
AR	2.37	2.39	−1%	356/500	73% (with larger data set)
CE	3.92	2.66	33%	120/500	87% (with σ_{th} increased to 11%)
FT	30.25	2.70	91%	32/500	60% (with larger data set)

models. The strategy of adding new data points based on the prediction uncertainty leads to better prediction results compared to random sampling, offering better prediction quality with little data labeling required. It was demonstrated that, when active learning is applied, an initial data set size of 100 is sufficient to get mean squared errors of the test data below 2% for all processes, while random sampling leads to standardized MSE_{test} that is 1.6% worse than that achieved with active learning. Thus, unless large databases of labeled data are already available for building surrogate models, following the active learning approach is favourable for generating databases. Furthermore, it was shown that uncertainty-based active learning estimates the quality of a prediction made by ALAI, which ensures reliable results during application. The application of ALAI in energy and process system optimization frameworks is expected to help researchers to improve both the computational time for result generation, as well as the availability of the process predictions, since the call of surrogates does not suffer from convergence issues commonly encountered with simulation software. Multiple surrogate models can be trained and optimized for predictions; therefore, an adaptive and flexible model creation process is enabled that–compared to training one type of surrogate–is able to generate high-performance surrogates for a wide range of simulation complexity. When integrating ALAI with the optimization framework, good prediction quality with average prediction errors below 1% can be expected even for complex models and the selected prediction uncertainty threshold of 8%. This work demonstrates that the application of ALAI for replacing simulation models in superstructure optimization reduces computational expense in optimization and improves convergence. Prediction performance and computational expense were both improved using the ALAI approach, and it was proven to be well-suited for a wide range of model complexity. For some models, the recorded time savings were marginal; however, ALAI was shown to be a valuable add-on for simulation-based superstructure optimization problems. Especially for complex simulation models with runtimes above 10 s that are integrated in an optimization framework, time savings of up to 50% are expected. For simpler models, it was shown that calling only one surrogate instead of continually comparing performance of several surrogates integrated in ALAI yields similar time savings, which would also be translated to computational expense.

Future work includes adding other surrogate models to ALAI for ensuring a good representation of any process and improving its flexibility. Furthermore, as the uncertainty is currently derived only from the mean of the standard deviation, other metrics should be included as options in the procedure. Lastly, it is foreseen to expand the developed method toward the prediction of optimization results rather than process conditions and thus saving time when performing multi-objective optimization of large energy systems.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

JG conceptualized and designed the presented methodology, performed the result generation and analysis and prepared the original draft of the manuscript. IK contributed to the conceptualization and the analysis of results as well as the article preparation. FM supervised the work and contributed to the conceptualization. All authors contributed to manuscript revision, read, and approved the submitted version.

FUNDING

This research has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 818011.

ACKNOWLEDGMENTS

The authors would like to thank Michel Lopez for his support regarding IT infrastructure used for this research. Furthermore, we are grateful for the valuable contributions Alexandre Cadillon and Davíð Oern Jensson made towards the surrogate model design.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., et al. (2015). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *Softw. available tensorflow.org*.

Aspentech (2019). *Aspen Plus - Aspen Technology Inc.*. Available at https://www.aspentech.com/en/products/engineering/aspen-plus (Accessed October 9, 2019).

Bhaskar, A. (2001). Principles of Optimal Design: Modeling and Computation P. Y. Papalambros and D. J. Wilde Second Edition. Cambridge University Press, the Edinburgh Building, Cambridge CB2 2RU, UK. 2000. 390pp. Illustrated. £27.95. ISBN 0-521-62727-3. *Aeronaut. J.* 105, 458–459. doi:10.1017/s0001924000012458

Biau, G., and Scornet, E. (2016). A Random forest Guided Tour. *TEST* 25, 197–227. doi:10.1007/s11749-016-0481-7

Bode, G., Thul, S., Baranski, M., and Müller, D. (2020). Real-world Application of Machine-Learning-Based Fault Detection Trained with Experimental Data. *Energy* 198, 117323. doi:10.1016/j.energy.2020.117323

Breiman, L. (2001). Random Forests. *Machine Learn.* 45, 5–32. doi:10.1023/a:1010933404324

Caballero, J. A., and Grossmann, I. E. (2008). An Algorithm for the Use of Surrogate Models in Modular Flowsheet Optimization. *Aiche J.* 54, 2633–2650. doi:10.1002/aic.11579

Cozad, A., Sahinidis, N. V., and Miller, D. C. (2014). Learning Surrogate Models for Simulation-Based Optimization. *Aiche J.* 60, 2211–2227. doi:10.1002/aic.14418

da Silva, I. N., Hernane Spatti, D., Andrade Flauzino, R., Liboni, L. H. B., and dos Reis Alves, S. F. (2016). *Artificial Neural Networks*. New York, NY: Springer Berlin Heidelberg.

Dai, Z., Álvarez, M., and Lawrence, N. (2017). "Efficient Modeling of Latent Information in Supervised Learning Using Gaussian Processes," in Advances in Neural Information Processing Systems 30 (NIPS 2017). Vol. 30, Long Beach, CA, United States, December 4–9, 2017.

Davis, E., and Ierapetritou, M. (2009). A Kriging Based Method for the Solution of Mixed-Integer Nonlinear Programs Containing Black-Box Functions. *J. Glob. Optim* 43, 191–205. doi:10.1007/s10898-007-9217-2

Douglas, J. M. (1988). *Conceptual Design of Chemical Processes*. New York: McGraw-Hill.

Fahmi, I., and Cremaschi, S. (2012). Process Synthesis of Biodiesel Production Plant Using Artificial Neural Networks as the Surrogate Models. *Comput. Chem. Eng.* 46, 105–123. doi:10.1016/j.compchemeng.2012.06.006

Fernandes, F. a. N. (2006). Optimization of Fischer-Tropsch Synthesis Using Neural Networks. *Chem. Eng. Technol.* 29, 449–453. doi:10.1002/ceat.200500310

Forrester, A. I. J., and Keane, A. J. (2009). Recent Advances in Surrogate-Based Optimization. *Prog. Aerospace Sci.* 45, 50–79. doi:10.1016/j.paerosci.2008.11.001

Fourer, R., Gay, D. M., and Kernighan, B. W. (1990). A Modeling Language for Mathematical Programming. *Manage. Sci.* 36, 519–554. doi:10.1287/mnsc.36.5.519

Gal, Y., and Ghahramani, Z. (2016). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, in Proceedings of the 33rd International Conference on International Conference on Machine Learning. (New York, NY, United States: JMLR), 48, 1050–1059.

Gassner, M., and Maréchal, F. (2009). Methodology for the Optimal Thermo-Economic, Multi-Objective Design of Thermochemical Fuel Production from Biomass. *Comput. Chem. Eng.* 33, 769–781. doi:10.1016/j.compchemeng.2008.09.017

Gerber, L., Fazlollahi, S., and Maréchal, F. (2013). A Systematic Methodology for the Environomic Design and Synthesis of Energy Systems Combining Process Integration, Life Cycle Assessment and Industrial Ecology. *Comput. Chem. Eng.* 59, 2–16. doi:10.1016/j.compchemeng.2013.05.025

GPy (2012). *GPy: A Gaussian Process Framework in python*. Available at http://github.com/SheffieldML/GPy (Accessed January 13, 2021).

Granacher, J., Kantor, I. D., Lopez, M., and Maréchal, F. (2021). "Self-learning Surrogate Models in Superstructure Optimization," in *Computer Aided Chemical Engineering. Vol. 50 of 31 European Symposium on Computer Aided Process Engineering*. Editors M. Türkay and R. Gani (Amsterdam, Netherlands: Elsevier), 439–444. doi:10.1016/b978-0-323-88506-5.50069-3

Hansen, K., Biegler, F., Ramakrishnan, R., Pronobis, W., von Lilienfeld, O. A., Müller, K.-R., et al. (2015). Machine Learning Predictions of Molecular Properties: Accurate Many-Body Potentials and Nonlocality in Chemical Space. *J. Phys. Chem. Lett.* 6, 2326–2331. doi:10.1021/acs.jpclett.5b00831

Hasan, M. M. F., Baliban, R. C., Elia, J. A., and Floudas, C. A. (2012). Modeling, Simulation, and Optimization of Postcombustion CO2 Capture for Variable Feed Concentration and Flow Rate. 1. Chemical Absorption and Membrane Processes. *Ind. Eng. Chem. Res.* 51, 15642–15664. doi:10.1021/ie301571d

Henao, C. A., and Maravelias, C. T. (2011). Surrogate-based Superstructure Optimization Framework. *Aiche J.* 57, 1216–1232. doi:10.1002/aic.12341

Huang, D., Allen, T. T., Notz, W. I., and Zeng, N. (2006). Global Optimization of Stochastic Black-Box Systems via Sequential Kriging Meta-Models. *J. Glob. Optim* 34, 441–466. doi:10.1007/s10898-005-2454-3

IBM (2017). *IBM ILOG CPLEX 12.7 User's Manual*. Incline Village, NV: IBM ILOG CPLEX Optimization Studio.

Jablonka, K. M., Jothiappan, G. M., Wang, S., Smit, B., and Yoo, B. (2021). Bias Free Multiobjective Active Learning for Materials Design and Discovery. *Nat. Commun.* 12, 2312. doi:10.1038/s41467-021-22437-0

Jablonka, K. M., Ongari, D., Moosavi, S. M., and Smit, B. (2020). Big-Data Science in Porous Materials: Materials Genomics and Machine Learning. *Chem. Rev.* 120, 8066–8129. doi:10.1021/acs.chemrev.0c00004

Jin, R., Chen, W., and Simpson, T. W. (2001). Comparative Studies of Metamodelling Techniques under Multiple Modelling Criteria. *Struct. Multidisciplinary Optimization* 23, 1–13. doi:10.1007/s00158-001-0160-4

Jones, D. R. (2001). A Taxonomy of Global Optimization Methods Based on Response Surfaces. *J. Glob. Optimization* 21, 345–383. doi:10.1023/a:1012771025575

Kantor, I., Robineau, J.-L., Bütün, H., and Maréchal, F. (2020). A Mixed-Integer Linear Programming Formulation for Optimizing Multi-Scale Material and Energy Integration. *Front. Energ. Res.* 8, 49. doi:10.3389/fenrg.2020.00049

Kermani, M. (2018). "Methodologies for Simultaneous Optimization of Heat, Mass, and Power in Industrial Processes," (Switzerland: EPFL). Ph.D. thesis.

Lookman, T., Balachandran, P. V., Xue, D., and Yuan, R. (2019). Active Learning in Materials Science with Emphasis on Adaptive Sampling Using Uncertainties for Targeted Design. *npj Comput. Mater.* 5, 1–17. doi:10.1038/s41524-019-0153-8

Maréchal, F., and Kalitventzeff, B. (1998). Process Integration: Selection of the Optimal Utility System. *Comput. Chem. Eng.* 22, S149–S156. doi:10.1016/s0098-1354(98)00049-0

McBride, K., and Sundmacher, K. (2019). Overview of Surrogate Modeling in Chemical Process Engineering. *Chem. Ingenieur Technik* 91, 228–239. doi:10.1002/cite.201800091

McCulloch, W. S., and Pitts, W. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bull. Math. Biophys.* 5, 115–133. doi:10.1007/bf02478259

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics* 21, 239–245. doi:10.1080/00401706.1979.10489755

Meinshausen, N. (2006). Quantile Regression Forests. *J. Machine Learn. Res.* 7, 983–999. doi:10.5555/1248547.1248582

Miller, D. C., Siirola, J. D., Agarwal, D., Burgard, A. P., Lee, A., Eslick, J. C., et al. (2018). "Next Generation Multi-Scale Process Systems Engineering Framework," in *Computer Aided Chemical Engineering. Vol. 44 of 13 International Symposium on Process Systems Engineering (PSE 2018)*. Editors M. R. Eden, M. G. Ierapetritou, and G. P. Towler (Amsterdam, Netherlands: Elsevier), 2209–2214. doi:10.1016/B978-0-444-64241-7.50363-3

Mirkouei, A., and Haapala, K. R. (2014). "Integration of Machine Learning and Mathematical Programming Methods into the Biomass Feedstock Supplier Selection Process." in Proceedings of the 24th International Conference on Flexible Automation & Intelligent Manufacturing, San Antonio, TX. (Lancaster, PA: DEStech Publications, Inc.), 443–450. doi:10.14809/faim.2014.0443

Mountraki, A. D., Benjelloun-Mlayah, B., and Kokossis, A. C. (2020). A Surrogate Modeling Approach for the Development of Biorefineries. *Front. Chem. Eng.* 2. doi:10.3389/fceng.2020.568196

Nascimento, C. A. O., Giudici, R., and Guardani, R. (2000). Neural Network Based Approach for Optimization of Industrial Chemical Processes. *Comput. Chem. Eng.* 24, 2303–2314. doi:10.1016/S0098-1354(00)00587-1

Nuchitprasittichai, A., and Cremaschi, S. (2013). Optimization of CO2 Capture Process with Aqueous Amines-A Comparison of Two Simulation-Optimization Approaches. *Ind. Eng. Chem. Res.* 52, 10236–10243. doi:10.1021/ie3029366

Palmer, K., and Realff, M. (2002). Metamodeling Approach to Optimization of Steady-State Flowsheet Simulations. *Chem. Eng. Res. Des.* 80, 760–772. doi:10.1205/026387602320776830

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine Learning in Python. *J. Machine Learn. Res.* 12, 2825–2830.

Pedrozo, H. A., Rodriguez Reartes, S. B., Chen, Q., Diaz, M. S., and Grossmann, I. E. (2020). Surrogate-model Based MILP for the Optimal Design of Ethylene Production from Shale Gas. *Comput. Chem. Eng.* 141, 107015. doi:10.1016/j.compchemeng.2020.107015

Peduzzi, E. (2015). "Biomass to Liquids," (Switzerland: EPFL). Ph.D. thesis.

Popovics, G., and Monostori, L. (2016). An Approach to Determine Simulation Model Complexity. *Proced. CIRP* 52, 257–261. doi:10.1016/j.procir.2016.07.072

Queipo, N. V., Pintos, S., Rincón, N., Contreras, N., and Colmenares, J. (2002). Surrogate Modeling-Based Optimization for the Integration of Static and Dynamic Data into a Reservoir Description. *J. Pet. Sci. Eng.* 35, 167–181. doi:10.1016/S0920-4105(02)00238-3

Quirante, N., Javaloyes, J., and Caballero, J. A. (2015). Rigorous Design of Distillation Columns Using Surrogate Models Based on Kriging Interpolation. *Aiche J.* 61, 2169–2187. doi:10.1002/aic.14798

Rasmussen, C. E., and Williams, C. K. I. (2006). Gaussian Processes for Machine Learning. *Adaptive Computation and Machine Learning*. Cambridge, Mass: MIT Press. OCLC: ocm61285753.

Rios, L. M., and Sahinidis, N. V. (2013). Derivative-free Optimization: a Review of Algorithms and Comparison of Software Implementations. *J. Glob. Optim* 56, 1247–1293. doi:10.1007/s10898-012-9951-y.00260

Sener, O., and Savarese, S. (2018). Active Learning for Convolutional Neural Networks: A Core-Set Approach, in Conference Track Proceedings of the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30–May 3, 2018. OpenReview.

Settles, B. (2012). Active Learning. *Synth. Lectures Artif. Intelligence Machine Learn.* 6, 1–114. doi:10.2200/S00429ED1V01Y201207AIM018

Shen, Y., Yun, H., Lipton, Z. C., Kronrod, Y., and Anandkumar, A. (2017). Deep Active Learning for Named Entity Recognition, in Proceedings of the 2nd Workshop on Representation Learning for {NLP}. Vancouver, Canada: Association for Computational Linguistics, 252–256.

Sikorski, J. J., Brownbridge, G., Garud, S. S., Mosbach, S., Karimi, I. A., and Kraft, M. (2016). Parameterisation of a Biodiesel Plant Process Flow Sheet Model. *Comput. Chem. Eng.* 95, 108–122. doi:10.1016/j.compchemeng.2016.06.019

Simpson, T., Mistree, F., Korte, J., and Mauery, T. (1998). "Comparison of Response Surface and Kriging Models for Multidisciplinary Design Optimization," in *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization* (American Institute of Aeronautics and Astronautics). doi:10.2514/6.1998-4755

Smith, M. (1993). *Neural Networks for Statistical Modeling*. New York, NY, USA: John Wiley & Sons.

Teske, L. S. (2014). "Integrating Rate Based Models into a Multi-Objective Process Design & Optimisation Framework Using Surrogate Models," (Switzerland: EPFL). Ph.D. thesis.

Tock, L., and Maréchal, F. (2014). Process Design Optimization Strategy to Develop Energy and Cost Correlations of CO2 Capture Processes. *Comput. Chem. Eng.* 61, 51–58. doi:10.1016/j.compchemeng.2013.10.011

Turton, R., Shaeiwitz, J. A., Bhattacharyya, D., and Whiting, W. B. (2018). *Analysis, Synthesis and Design of Chemical Processes. Pearson Education*. 5th Edn. Boston: Prentice Hall.

Ulrich, G. D., and Vasudevan, P. T. (2003). *Chemical Engineering Process Design and Economics: A Practical Guide*. 2 edn. Boca Raton, Florida: CRC Press.

Wernet, G., Bauer, C., Steubing, B., Reinhard, J., Moreno-Ruiz, E., and Weidema, B. (2016). The Ecoinvent Database Version 3 (Part I): Overview and Methodology. *Int. J. Life Cycle Assess.* 21, 1218–1230. doi:10.1007/s11367-016-1087-8

Zhang, X., Song, Y., Wang, G., and Bao, X. (2017). Co-electrolysis of CO2 and H2O in High-Temperature Solid Oxide Electrolysis Cells: Recent advance in Cathodes. *J. Energ. Chem.* 26, 839–853. doi:10.1016/j.jechem.2017.07.003

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

NOMENCLATURE

Abbreviations

ALAI active learning artificial intelligence

ANN artificial neural network

DL deep learning

LCI life cycle inventory

LHS latin hypercube sampling

MSE mean squared error

Number Sets

$\boldsymbol{i} \in \boldsymbol{I}$ iteration in designing ALAI

$\boldsymbol{j} \in \boldsymbol{J}$ optimization calls

$\boldsymbol{t} \in \boldsymbol{t}$ system states

$\boldsymbol{u} \in \boldsymbol{U}$ units in the superstructure

Other Symbols

$\hat{\boldsymbol{\sigma}}$ prediction uncertainty

$\hat{\boldsymbol{y}}$ predictions

$\boldsymbol{\sigma}_{\mathbf{th}}$ threshold of prediction uncertainty

$\boldsymbol{C}_{\mathbf{model}}$ model complexity

$\boldsymbol{C}_{\mathbf{units}}$ overall unit complexity

$\boldsymbol{C}_{\boldsymbol{u}}$ complexity of unit u

$\boldsymbol{d}$ design variables of ALAI

$F_{\boldsymbol{u}}^{\mathbf{min/max}}$ lower/Upper bound of unit multiplication factor

$\boldsymbol{f}_{\mathbf{mult}}(\boldsymbol{u})$ unit multiplication factor

$\boldsymbol{N}_{\mathbf{equ}}$ number of equations

$\boldsymbol{N}_{\mathbf{react}}$ number of reactions

$\boldsymbol{N}_{\boldsymbol{s},\mathbf{in/out}}$ number of streams in/out

$\boldsymbol{N}_{\boldsymbol{u},\mathbf{in/out}}$ number of streams in/out of unit u

$\boldsymbol{N}_{\mathbf{var}}$ number of variables

$\boldsymbol{n}_{\mathbf{total}}$ number of samples created when designing ALAI

$\boldsymbol{V}$ data set of features and labels of ALAI

$\boldsymbol{X}$ features of ALAI

$\boldsymbol{Y}$ labels of ALAI

$\boldsymbol{Y}_{\boldsymbol{u}}$ bound on unit installation, indicates if unit is considered

$\boldsymbol{Y}_{\mathbf{flex}}$ binary stating if mathematical equations are used in unit

$\boldsymbol{y}_{\mathbf{use}}$ binary decision variable on unit installation