sensors

MDPI

Article

Industrial Soft Sensor Optimized by Improved PSO: A Deep Representation-Learning Approach

Alcemy Gabriel Vitor Severino , Jean Mário Moreira de Lima and Fábio Meneghetti Ugulino de Araújo *

Computer Engineering and Automation Department, Federal University of Rio Grande do Norte,
3000 Senador Salgado Filho Avenue, Natal 59078-970, RN, Brazil
* Correspondence: meneghet@dca.ufrn.br; Tel.: +55-84-98818-5127

Abstract: Soft sensors based on deep learning approaches are growing in popularity due to their ability to extract high-level features from training, improving soft sensors' performance. In the training process of such a deep model, the set of hyperparameters is critical to archive generalization and reliability. However, choosing the training hyperparameters is a complex task. Usually, a random approach defines the set of hyperparameters, which may not be adequate regarding the high number of sets and the soft sensing purposes. This work proposes the RB-PSOSAE, a Representation-Based Particle Swarm Optimization with a modified evaluation function to optimize the hyperparameter set of a Stacked AutoEncoder-based soft sensor. The evaluation function considers the mean square error (MSE) of validation and the representation of the features extracted through mutual information (MI) analysis in the pre-training step. By doing this, the RB-PSOSAE computes hyperparameters capable of supporting the training process to generate models with improved generalization and relevant hidden features. As a result, the proposed method can generate more than 16.4% improvement in RMSE compared to another standard PSO-based method and, in some cases, more than 50% improvement compared to traditional methods applied to the same real-world nonlinear industrial process. Thus, the results demonstrate better prediction performance than traditional and state-of-the-art methods.

Keywords: particle swarm optimization; soft sensors; deep learning; stacked autoencoders; mutual information

check for updates

Citation: Severino, A.G.V.; de Lima, J.M.M.; de Araújo, F.M.U. Industrial Soft Sensor Optimized by Improved PSO: A Deep Representation-Learning Approach. *Sensors* **2022**, *22*, 6887. https://doi.org/10.3390/s22186887

Academic Editor: Viorel Minzu

Received: 1 August 2022
Accepted: 16 August 2022
Published: 13 September 2022

1. Introduction

Numerous key-quality process variables are hard and high-cost to measure in real-time in complex industrial processes, specifically in the oil and chemical industries. In the lack of online measurements for such critical variables, efficient monitoring and control strategies may not be available. In such cases, inference approaches like soft sensors may surpass the above-cited problem [1]. Soft sensors can estimate the hard-to-measure variables using secondary variables, which are easy and low-cost to measure [2]. Several proposed methods have designed soft sensors, and models based on artificial intelligence techniques have succeeded in various applications, including industrial scenarios. Principal component regression (PCR), Support vector machine (SVM), Gaussian Process Regression (GPR), Partial Least Square (PLS), and Artificial Neural Network are among the most thriving and used methods [3–13].

Generally, in cases of building virtual sensors to measure quality variables in processes, there is not a lot of labeled data, but there is plenty of unlabeled data. In this case, semi-supervised methods are more promising alternatives than traditional methods, which demonstrate unsatisfactory performance when they have a limited amount of labeled data [14]. The extensive volume of unlabeled data stores latent information, which, when used correctly, can improve model reliability and prediction performance [13]. Deep learning strategies are increasingly being used in the implementation of semi-supervised methods [15]. A deep network architecture, known as a stacked autoencoder (SAE), which has its weights calculated by unsupervised pre-training and applied to the supervised fine-tuning step, is successfully used in many soft sensor designs applied in industrial processes [16–20].

However, one of the difficulties of deep neural networks applications such as SAE is the definition of your hyperparameters: batch size, learning rate, step hidden features, among others. Evaluating the many possible hyperparameter configurations would require a high cost of time. Thus, the task of defining hyperparameters is an optimization problem.

Exact algorithms are methods used to find an optimal solution to a given problem, in which no time limits are imposed on the search process. Consequently, exact algorithms require a high computational effort. An alternative to exact algorithms are meta-heuristics. These algorithms are inspired by nature, i.e., natural phenomena and/or physical laws. Meta-heuristics propose to combine basic heuristic methods to effectively explore a search space. For this reason, several meta-heuristics have been applied to the hyperparameters' definition of deep neural networks, such as Bayesian Optimization (BO) [21,22], Genetic Algorithm (GA) [21,23], Harmony Search (HS) [24], Whale Optimization Algorithm (WOA) [25], and Particle Swarm Optimization (PSO) [26].

The work of [21] used four hyperparameter optimization methods to design a rotation angle estimator based on an artificial neural network: Random Search (RS), Hyperband (HB), BO, and GA. Among the methods, the BO and GA metaheuristics showed better results in selecting the hyperparameters in vast search space and strongly nonlinear problems. Ref. [22], meanwhile, optimized the hyperparameters and structure of a convolutional neural network (CNN) applied in a data-driven intelligent fault diagnosis technique for rotating machines. The results demonstrate the efficiency of BO in optimizing the hyperparameters and the network structure when the objective function is time, which is computationally expensive in the cases of CNNs. In [23], the authors proposed a strategy of hyperparameter optimization based on Non-dominated Sorting Genetic Algorithm-II (NSGA-II) in the training process of a deep learning model. The obtained results showed improved generalization error and lowered overfitting incidence ranting. Furthermore, the proposal [24] applies an HS metaheuristic to optimize the hyperparameters of a 1D CNN model, aiming for accuracy improvements in pattern recognition. The improved model has shown a better precision rate than the non-optimized CNN model. Ref. [25] presents the WOA metaheuristic for optimizing the hyperparameters of a neural network. The search behavior of humpback whales inspires the method. WOA found a good set of hyperparameters in a shorter period when compared to the GS method while being of similar quality to the more straightforward RS method. Ref. [26] proposes a novel soft sensing approach based on semi-supervised ensemble learning. A novel online model adaptation criterion approach accurately describes the relationships among samples and local models and can provide higher mensuration exactness. The parameters of the presented technique are completed automatically by the PSO method. The simulation outcomes reveal the significance of the suggested process in dealing with nonlinear regression problems.

The evaluation functions of the previously-mentioned methods only regard information based on error indices between the output generated by the models and the actual output of the systems as mean squared error. However, the evaluation functions do not consider the representation relevance of the features extracted in the pre-training stage, an essential metric for soft sensors based on deep learning. This work proposes the RB-PSOSAE, a Representation-Based Particle Swarm Optimization with a modified evaluation function to optimize the hyperparameter set of an SAE-based soft sensor. The evaluation function considers the mean square error (MSE), as in previous works, and the relevance of the information extracted from the unlabeled data set through mutual information (MI) analysis in the pre-training step. Doing so enhances PSO to compute hyperparameters capable of generating models with improved generalization and relevant hidden features, and then better performance than traditional-based PSO.

The main contributions of this research are as follows:

1. Present an improved PSO for the automatic adjustment of hyperparameters of deep neural networks based on the relevance of extracted representations;
2. Carry out the extraction of representative features through the analysis of mutual information used in the PSO evaluation function;
3. Improve the performance of the SAE model used for feature extraction in the unsupervised learning stage;
4. Obtain a neural model with relevant features generated from an optimal combination of hyperparameters using the unlabeled data.

The contributions mentioned above have been demonstrated to be acceptable and successful for the automatic adjustment of hyperparameters for an industrial plant of a debutanizer column. The work sections are as follows: Section 2 introduces theoretical fundaments. Section 3 explains the proposal in details. Furthermore, Section 4 presents the obtained results. Finally, Section 5 shows the conclusions and future perspectives of this work.

2. Preliminaries

2.1. Autoencoders

Autoencoder (AE) is an artificial neural network composed of two components: the encoder and the decoder [27]. The encoder aims to map its input into a low-dimensional, high-level, meaningful representation. Conversely, the decoder receives the encoder's output and tries to rebuild the original input. In this process, the autoencoder can learn high-relevant features from input data in reconstructing the input by using extracted hidden features. Figure 1 illustrates an AE architecture [13].

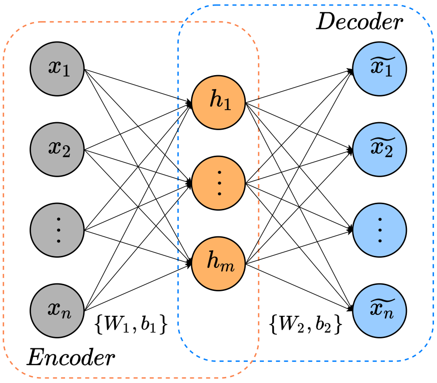

Figure 1. Basic AE schematic.

The encoder receives input $\mathbf{x} = [x_1, x_2, \cdots, x_n]^T \in R^n$ and transforms it into a low-dimensional hidden features $\mathbf{h} = [h_1, h_2, \cdots, h_m]^T \in R^m$. Additionally, the decoder computes the previous-obtained hidden features to try to map out the input data. Equations (1) and (2) characterize the above-cited encoder and decoder operations:

$$\mathbf{h} = f(\mathbf{W}_e\mathbf{x} + \mathbf{b}_e), \tag{1}$$

$$\widehat{\mathbf{x}} = g(\mathbf{W}_d\mathbf{h} + \mathbf{b}_d), \tag{2}$$

where $W_e \in R^{n\times m}$, $b_e \in R^m$, $W_d \in R^{m\times n}$ and $b_e \in R^n$ are the weight matrices and bias of the encoder and decoder, respectively. Terms f and g are the commonly-used activation functions sigmoide or ReLU [28]. Figure 2 demonstrates the schematic diagram of an SAE.

The mean square error (MSE) among $\mathbf{x}$ and $\widehat{\mathbf{x}}$ represents the training loss function of AE as Equation (3) illustrates. The learning process adjusts the parameters set (W_e, b_e, W_d, b_d) to minimize the reconstruction error:

$$J_{AE} = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\|\widehat{x}_i - x_i\|^2\right) \tag{3}$$

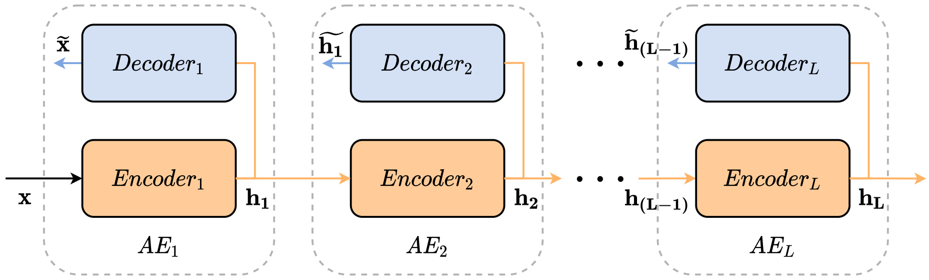

Figure 2. Stacked Autoencoders schematic diagram.

When several AE are stacked, it builds a deep structure named stacked autoencoder (SAE), which is able to learn high-level features since each AE is an SAE layer. Basically, an SAE architecture uses the previous layer output as input to feed the next layer. Training such a model normally takes two stages: the unsupervised pre-training and the supervised fine-tuning. In the first stage, layer-by-layer pre-training tries to minimize the reconstruction loss function as Equation (3) illustrates. On the other hand, the second phase fine-tunes all SAE parameters by minimizing the prediction error [29,30].

An SAE model uses unlabeled and labeled data samples to develop semi-supervised soft sensors, but the model does not necessarily learn high-level representations for soft-sensing purposes. Naturally, unsupervised pre-training does not regard targeted-output data, which might degrade the prediction's performance even when a successful fine-tuning performs [31]. In addition, an SAE model has several hyperparameters to be chosen, and each one of them can impact the model's performance and extracted-features quality. Furthermore, the optimization of an SAE's hyperparameters regarding the relevance of the extracted hidden features is critical to build a suitable SAE-based soft sensor. In this work, an enhanced particle swarm optimization, which considers features' representation relevance, optimizes an SAE's hyperparameters.

2.2. Particle Swarm Optimization

PSO was inspired by nature, based on the social characteristics of bands of birds and schools of fish in search of a nest or food [32]. PSO is widely applied in solving optimization problems [33] due to the characteristics of few parameters, simple formulas, easy implementation, and good convergence speed. Recent advances in the application of PSO are emerging in the optimization of hyperparameters of SAEs [34–36]. According to [37], the PSO is similar to the GA, as the system is initialized with a population of random candidate solutions, here called a swarm. However, this population presents differences because a velocity $\mathbf{v}_i$, Equation (4), is assigned for each potential solution, and the potential solutions, called particles, are guided through the solution space. Each particle has a position $\mathbf{x}_i$, Equation (5), within the search space. The best solution found by the i-th particle up to the iteration k, $\boldsymbol{pbest}_i$, is linked to this position. Another critical variable in the algorithm execution is the best global solution, $\boldsymbol{gbest}$, which is the best solution found by the i-th swarm up to the iteration k. The velocity updating of each particle through $\boldsymbol{pbest}$ and $\boldsymbol{gbest}$ and, later, its position is the base of PSO operation:

$$\mathbf{v}_i^{k+1} = \omega \mathbf{v}_i^k + r_1\phi_1(\boldsymbol{pbest}_i^k - \mathbf{x}_i^k) + r_2\phi_2(\boldsymbol{gbest} - \mathbf{x}_i^k), \tag{4}$$

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \mathbf{v}_i^{k+1}, \tag{5}$$

in which ω is the inertia factor, responsible for promoting a balance between global and local exploration. The inertia factor is related to the step of the method. If its value is too high, particles can pass through acceptable solutions without visiting them; on the other hand, if its value is too low, the particles may not sufficiently explore places that have acceptable solutions. The terms ϕ_1 and ϕ_2 represent the degree of confidence of the particle in the best solution found by it, $\boldsymbol{pbest}_i$, and in the best solution found by the swarm, $\boldsymbol{gbest}$. If their values are too low, the particles will "fly" for more distant routes towards the target region, or if their values are too high, the particles will make abrupt movements towards the potential region. r_1 and r_2 are random numbers that range between 0 and 1.

3. The Proposed Method

This section details the proposal step-by-step. First, the proposed representation-based PSO performs hyperparameter optimization in the unsupervised pretraining of an SAE model, generating an optimized SAE named RBPSO-SAE. In the next stage, an LSTM structure couples with the RBPSO-SAE to proceed with the supervised fine-tuning.

3.1. Data Preprocessing

As the first step, the dataset $\{\mathbf{X}, \mathbf{Y}\}$ is standardized into a range of $[0, 1]$. Such preprocessing can improve the overall stability of the model. Unlabeled $\{\mathbf{X}^U\}$ and labeled $\{\mathbf{X}^L, \mathbf{Y}\}$ data compose the dataset with ratios of 90% data and 10%, respectively. This scenario emulates real industrial scenarios where labeled is scarce, but the number of unlabeled samples is abundant. The large set of non-labeled data can hide relevant features about the process that traditional techniques do not exploit. Then, the unsupervised pretraining uses the unlabeled data to train the RBPSO-SAE models, aiming extraction of meaningful features.

The training set $\{\mathbf{X}^L, \mathbf{Y}\}_{Tr}$ represents 40% of the labeled set, and a total of 10% of the labeled set forms the validation set $\{\mathbf{X}^L, \mathbf{Y}\}_V$, and, finally, the testing set $\{\mathbf{X}^L, \mathbf{Y}\}_{Te}$ takes 50% from the labeled set. The supervised fine-tuning uses the three above-cited labeled subsets to train, validate, and test the entire deep architecture composed by the RBPSO-SAE coupled to an LSTM.

3.2. Representation-Based PSO

A regular AE aims to reconstruct the input data at its output, and in this process, it learns meaningful representations of the input data. In the learning process, a regular AE treats all available data similarly. However, it is not true that all variables are equally relevant to building AEs for soft sensing purposes. Once irrelevant information is on an AE-based soft sensor, it can damage prediction performance.

This work presents a representation-based PSO to optimize the hyperparameters for SAE's pretraining. The proposed PSO evaluates two critical points: the test mean square error (MSE), which measures the capacity of reconstructing the inputs, and the mutual information (MI) between features and targeted outputs to analyze how relevant extracted features are.

Choosing hyperparameters is one of the most complex parts of training deep learning models due to their magnitude, variables' volume, and underlying correlation. In addition, there is not a formal method to do it beyond empirical methods such as random and grid search, which does not guarantee the optimal hyperparameter set. Therefore, the representation-based PSO tries to balance MSE and MI to get an optimized hyperparameter set capable of generating an enhanced AE with relevant features and, then, better adapted for soft-sensing.

The following steps detail the proposal:

Step 1. MI analysis evaluates the nonlinear relationship between each extracted feature and target outputs. Suppose the calculated MI is not greater than an early-defined threshold value, representing the minimum required relevance. In that case, the computed feature x_i is irrelevant to inferring the desired outputs as follows:

$$MI(x_i, y) \leq th, \tag{6}$$

where th is the threshold value.

As the first step, the proposal determines the MI threshold value. A uniform distribution generates a 1000 random vectors with values in the range of $[0, 1]$. MI analysis among each generated arbitrary vector and the targeted output is computed. After that, calculated MI values are ordered in a descending way, and the 50th value points to the th. As a result, when computed MI values are greater than th, the confidence level is 95%.

Step 2. The second step specifies the hyperparameters. The selected training hyperparameters are the batch size BS, the learning rate LR, and the number of hidden features HF. As an AE has an intrinsic characteristic to extract hidden features by reconstructing its input, reducing the number of hidden features by each layer boosts AE ability and representations as the model gets deep. Therefore, this proposal does not use a HF for each layer but one HF that decreases for each model layer proportionally.

Another concern is that the range of hyperparameters needs to be defined. There is no formal method to set up the hyperparameters to define their searching space. Determined inferior and superior range limits have to guarantee that PSO can use all relevant-available searching space.

Step 3. The third step defines the early-needed PSO parameters for running the optimization attempts: the number of swarms N_{SW}, the number of particles N_{pr}, the inertial factor ω, and confidence parameters Φ_1 and Φ_2. The N_{SW} and N_{pr} relate to the number of evaluated models. The higher these values, the greater the time spent and the computational cost. For this reason, choosing these values is a task where balance among the number of models and performance matters. The inertia factor ω relates to the search step of the optimization algorithms. A low inertia factor will result in a lower speed of the particles, that is, a smaller displacement. In comparison, a high value will cause a higher speed of the particles, consequently a more expressive displacement in the search space. Finally, Φ_1 and Φ_2 represent the particle's confidence in itself and the best solution found by the group, respectively. These values relate to the degree of exploration of the search space, influencing particles' convergence speed to find the best solution.

Step 4. Step four explains the representation-based PSO as follows:

Step 4.1. Once the PSO parameters are defined, we randomly generate a set of initial hyperparameters, the swarm S_0 with $P_{S_0} = \{P_0, P_1, ...P_{N_{pr}}\}$ particles set regarding hyperparameters' ranges. Each particle of P_{S_0} represents a set with values of BS, LR, and HF that set up the hyperparameters for the SAE's pretraining.

Step 4.2. By applying the first particle of P_{S_0} set, the pretraining of the first stacked autoencoder gets started. The process applies all available particles in P_{S_0} until the training process of N_{pr}-th model ends. In sequence, the proposal evaluates the set of representation-based PSO stacked autoencoders (*RBPSO-SAE*).

Step 4.3. This work evaluates each of the obtained *RBPSO-SAE* models regarding their ability to extract features and how relevant those features are for the main goal of the model, which is virtual sensing tasks. First, the proposal calculates the MSE between real and targeted outputs from the test set, evaluating the model's performance. Second, MI analysis among target outputs and the model's output performs the representation-based evaluation. All of the RBPSO-SAE models receive the labeled dataset $\{X^L, Y\}$ as input, then they generate a representation-based output Φ, which are the high-level extracted features from the input data. The MI analysis of the nonlinear relationship among features of Φ and the targeted-output values Y regarding Equation (6). The higher the MI value is, the greater the relevance of the analyzed feature to estimate desired outputs. By using MSE and the mean of MI values, MI_{mean}, the approach presents a fitness function (Figure 3) used to measure the appropriate level of each model as follows:

$$fitness = \alpha * MSE + \frac{1}{\beta * MI_{mean}} \tag{7}$$

where α and β are values to tune the importance level of MSE and MI values in obtaining an optimized model. The lower the value of the fitness function, the better the solution. The proposed method updates the best global solution $gbest$ and the $pbest$, which is the best solution found by the i-th particle up to the iteration k.

The above-described representation-based PSO process repeats itself, generates a new swarm, and evaluates the new particle-generated models.

Step 5. When the representation-based PSO finishes, the best generated global model composes the soft sensor's architecture. An LSTM structure couples to the best-found

RBPSO-SAE to accomplish the regression task. An LSTM model has the intrinsic ability to handle time-series input and its dynamics. As an industrial process is highly nonlinear and dynamic, LSTM suits soft-sensing purposes well. The LSTM receives the highly in-depth and relevant features from the RBPSO-SAE and then estimates the targeted-output values.

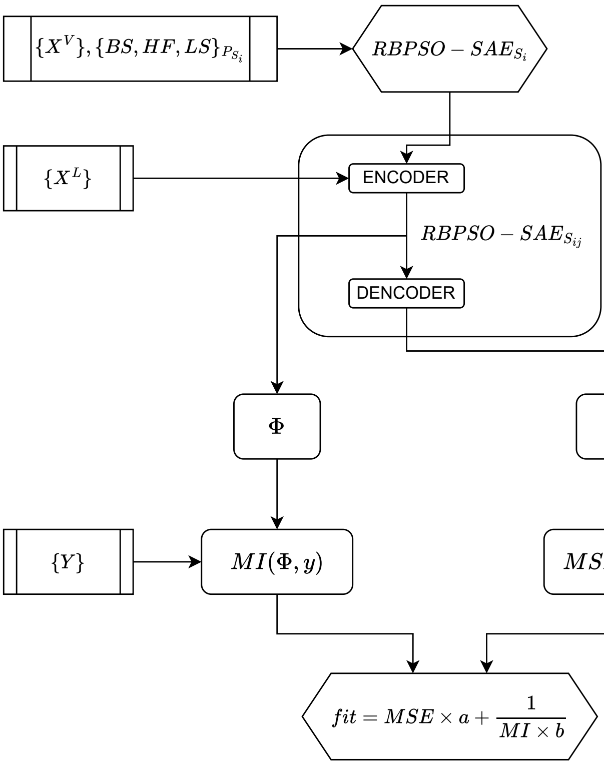

Figure 3. Fitness function flowchart.

4. Case Studies and Results

The RBPSO-SAE-based soft sensor evaluation proceeds through an industrial-based plant debutanizer column case study. For comparison, this work utilizes the following models:

1. Deep learning-based methods: SAE with grid search (GS-SAE) and SAE with random search (RS-SAE);
2. Deep learning-based method with PSO optimization: SAE with PSO tuning hyperparameters through MSE only (PSO-SAE);
3. Proposed relevant representation-based PSO soft sensor model: RBPSO-SAE.

The root-mean-square error ($RMSE$) and coefficient of determination (R^2) are the chosen metrics for comparing the above-cited methods' prediction efficiency:

$$RMSE = \sqrt{\frac{1}{N_{Ts}} \sum_{i=1}^{N_{Ts}} (\hat{y}_i - y_i)^2}, \tag{8}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N_{Ts}} (\hat{y}_i - y_i)^2}{\sum_{i=1}^{N_{Ts}} (y_i - \overline{y}_i)^2}. \tag{9}$$

Equations (8) and (9) compute $RMSE$ and R^2, respectively. The terms y_i and $\hat{y}_i$ represent the real and estimated output, while the $\overline{y}$ represents the mean value of actual outputs. The N_{Ts} represents the number of samples in the testing set. The RMSE indicates the error between targeted and estimated values. It is a metric usually used to evaluate soft sensor performance, and it quantifies the deviation between predicted and actual values in a squared error sense [38]. Therefore, RMSE quantifies reliability and the prediction performance [39]. Beyond that, predicting quality variables has inherent uncertainty. The standard deviation (*SD*) of RMSE is the adopted metric to measure the uncertainty range of the attained results over different runs [40].

The R^2 represents a correlation between predicted and actual outputs, in the form of a variance value over the desired outputs, where a high value represents better performance and higher reliability of the model [41].

4.1. Industrial Debutanizer Column Process

The debutanizer columns are regular devices that process desulfurization and naphtha cracking in oil and gas refineries [42]. Withdrawing propane (C5) and butane (C4) from naphtha steam is the primary goal of such an apparatus. Consequently, the lower the amount of butane, the better the quality of the naphtha end products. However, no physical sensors can measure the amount of butane in real-time. One solution is to use soft sensors to estimate the butane concentration simultaneously. Figure 4 illustrates the debutanizer column and its devices. The squares containing gray circles represent the hardware sensors that measure process variables such as flow, pressure, and temperature. The purpose of the debutanizer column used is to remove C3 and C4 from naphtha steam. Reducing the C4 concentration increases the quality of the final product located at the bottom of the debutanizer column. Gas chromatographs measure C4 concentration. However, due to a long measurement interval, they cannot provide real-time C4 concentrations for monitoring and control purposes. As explained earlier, soft sensors can estimate real-time measurements unavailable from physical sensors, such as C4 concentration. In [43], the dataset and more process details are available.

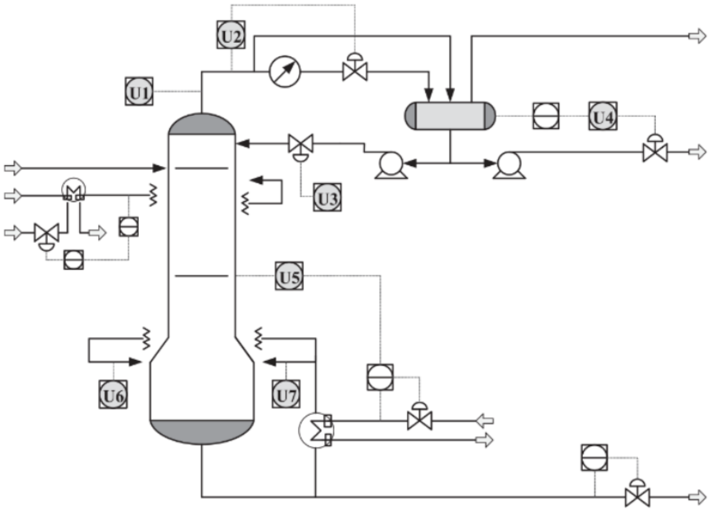

Figure 4. Schematic representation of the debutanizer column process [13].

Table 1 lists the process variables present in the debutanizer column process. The study-case debutanizer offers 2384 data samples for each process variable, with a sampling time T_s = 6 min. However, in a real distillation column scenario, not all of the data are labeled. Therefore, the proposal uses only a tiny part of the data samples as labeled in order to reproduce real scenarios where labeled data are scarce. The fine-tuning stage utilizes 240 samples, representing only 10% of the total data. Then, the non-labeled dataset has 2144 data samples, representing 90% of available data applied in the unsupervised pretraining.

As a regular dynamic system, the study-case debutanizer column outputs are a product of current inputs and past outputs. A feature engineering approach can handle the dynamicity of the process: past input and output values incorporate the current input. For the proposed model, the input blends to $X = [\mathbf{u}(t), ..., \mathbf{u}(t-d_x), \mathbf{y}(t-1), ..., \mathbf{y}(t-d_y)]$, where u and y are inputs and outputs, and d_x and d_y time-delay of inputs and outputs, respectively. This work employs $d_x = d_y = 6$.

Table 1. Description of debutanizer column process variables.

Variable	Variable Description	Unit
u1	Top temperature	°C
u2	Top pressure	kg/cm^2
u3	Reflux flow	m^3/h
u4	Flow to next process	m^3/h
u5	Sixth tray temperature	°C
u6	Bottom temperature A	°C
u7	Bottom temperature B	°C
Output	Butane C4 content in IC5	-

4.2. RBPSO-SAE

To get started with RBPSO-SAE strategy, the proposal set has a total of 30 swarms with 10 particles each, thus adding up to 300 model evaluations. It is important to note that this does not necessarily mean that this approach trained 300 distinct models since particles with the same sets of hyperparameters are present in all swarms. A value of ω is equal to 0.01. This value is related to the search step of the optimization algorithms, so, interestingly, its value is neither high nor low considering the magnitudes of the hyperparameters. Furthermore, all hyperparameters use the same ω. The values 1.2 and 2.4 were set for ϕ_1 and ϕ_2, respectively. The chosen set-values mean that the particle has twice as much confidence in the best solution found by the swarms as in the best solution found by itself. Finally, the values that the hyperparameters can take belong to the ranges $[0.0005, 0.0100]$, $[10, 500]$, and $[2, 15]$ for learning rate (LR), batch size (BS), and hidden features (HF), respectively. However, they were normalized to the range $[0, 1]$ so that only one value of ω is necessary to set. Since all the parameters are properly set, the proposed RBPSO-SAE approach starts to build SAE's model with optimized hyperparameters based on the mean square error between estimated and actual outputs, and the MI-based relevance of extracted features.

Figure 5 illustrates the behavior of *g-best* particle, which means the best-obtained SAE model through the generated swarms, regarding the fitness criteria: mean square error and mean of MI values between extracted features and targeted output. By observing the graph of the evolution of the MSE, the mean, and the standard deviation of the MI, we can notice the following: There is a sharp drop in the MSE value between the first and the fifth swarm, stabilizing between the sixth and the sixteenth swarm. Finally, it decreases from the seventeenth swarm and returns to stability at the twenty-sixth. The mean MI value starts stable, but there is a noticeable increase between the third and fourth swarm. Soon after, it remains stable between the fifth and the sixteenth swarm and then acquires a rising behavior until the twenty-sixth swarm. MI's mean and standard deviation values start stable, showing a decrease between the second and third swarms. Next, its value remains stable between the fourth and fifteenth swarm. Finally, it shows a falling behavior between the sixteenth and twenty-first swarm, and then remains stable. The increase in the mean MI value and the decrease in the MSE value and MI standard deviation along the swarm's evolution imply improvements in the projected SAE performance. A lower MSE value of the validation implies a better generalization. As well as a higher value of the mean of the MI, we obtain more representative features. A smaller value of the MI standard deviation means minor variation in the quality of the features extracted from the data. Thus, the previous graph proves the efficiency of enhanced-proposed PSO application in adjusting hyperparameters considering its evaluation of MI.

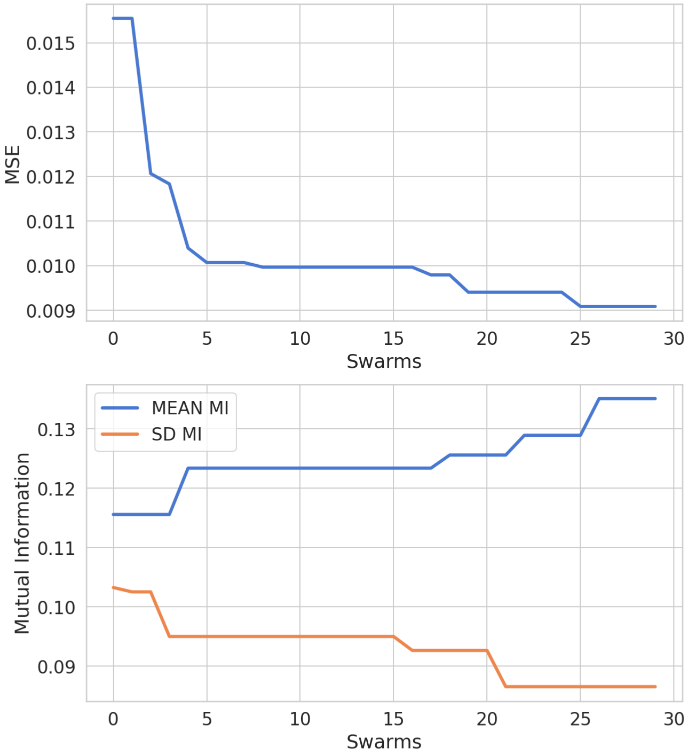

Figure 5. MSE, mean, and sd MI between input representations and output variables for the debutanizer column.

Table 2 compares the prediction performance of the models obtained by fitting the hyperparameters found by the algorithms applying traditional search methods of RSSAE and GSSAE, applying the PSOSAE meta-heuristic, and applying the proposed search method RS-PSOSAE. The algorithms that use the traditional search methods, RSSAE and GSSAE, obtain the worst results compared to the other methods. However, when using search methods that consider past results to find new solutions, the algorithms that apply metaheuristics have better results than traditional search methods. RB-PSOSAE obtains an improved result compared to the PSOSAE method, which is explained due to PSOSAE considering only MSE information in its evaluation function. The better result of the RB-PSOSAE happens because its evaluation function considers information from MSE and MI. Thus, only relevant representations are present in its acquired knowledge, making it more suitable for soft sensor applications. In addition, the prediction performance of the other methods was tested with the same debutanizer column process employed in this paper. Looking at Table 2, which contains the qualitative comparison of the methods, RB-PSOSAE showed the best result. PSOSAE, GSSAE, and RSSAE are methods that only consider the MSE in their evaluations. However, RB-PSOSAE, besides considering the MSE, also evaluates the searched solutions through MI analysis, explaining its improved performance. Furthermore, RB-PSOSAE has the lowest standard deviation (SD) of RMSE, which points to its stability under uncertain conditions. RB-PSOSAE, when compared to PSOSAE, showed a 16.4% improvement in RMSE value. Meanwhile, compared to GSSAE and RSSAE, it showed an improvement in RMSE values of 54.7% and 49.4%, respectively.

Table 2. Prediction performance of debutanizer column soft-sensor models.

Model	$RSME \pm SD$	R^2
RSSAE	0.050626 ± 0.0034	0.929491
GSSAE	0.056583 ± 0.0041	0.902333
PSOSAE	0.030635 ± 0.0023	0.974181
RB-PSOSAE	0.025604 ± 0.0014	0.981965

Figure 6 demonstrates the prediction results from the test dataset using parity plots. Again, RB-PSOSAE showed higher accuracy, as expected, compared to the other methods. In addition, Figure 7 illustrates the relative prediction errors with boxplots of the four methods, RB-PSOSAE, PSOSAE, GSSAE, and RSSAE, in descending order of performance. The box edges indicate the 25th and 75th percentiles, while the red mark in the center represents the median value within each box. The larger the box's width, the more dispersed the prediction errors. The upper and lower whiskers represent maximum and minimum values. The narrower box range of RB-PSOSAE indicates better prediction performance among the four compared methods mainly because RB-PSOSAE, through its evaluation function, can select the most relevant representations, extract nonlinear features, and handle process dynamics. Nonetheless, there are uncertainties to be considered in industrial processes [13,44], so Figure 7 shows some outlines represented individually by red dots. Inadequate initial parameters and discrepant values result in improper soft sensors with high relative prediction error. In contrast, acceptable soft sensors obtain stationary relative prediction errors for the same data set. Thus, stationary error values show the robustness of the model [13,45]. Therefore, the soft sensor designed by the RB-PSOSAE achieved better performance, reliability, and robustness.

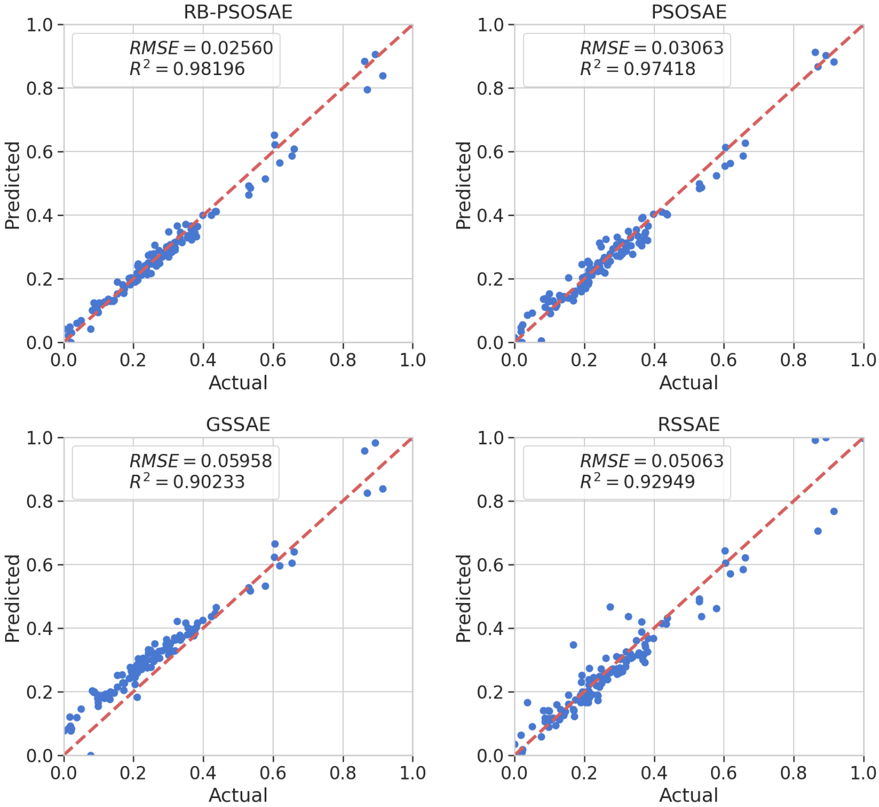

Figure 6. Real values of butane content and the predicted values using RB-PSOSAE, PSOSAE, GSSAE, and RSSAE search methods.

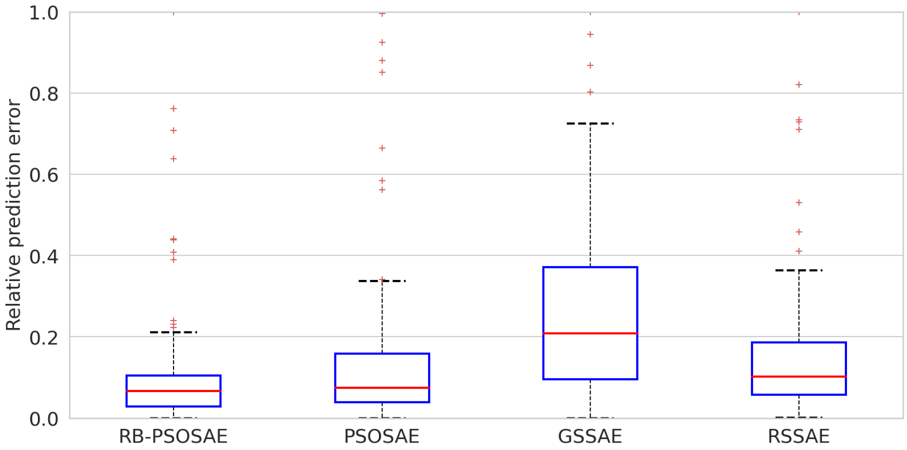

Figure 7. Relative prediction error of testing results for the debutanizer column process using RB-PSOSAE, PSOSAE, GSSAE, and RSSAE search methods.

5. Conclusions

An improved PSO based on representational learning has been proposed and tested for automatic tuning hyperparameters of deep neural networks. The RB-PSOSAE combines the search strategies of the PSO meta-heuristic, high-level feature extraction, and mines relevant representations in the SAE layers through MI analysis. In the PSOSAE method, particle swarms search for hyperparameter sets of an SAE using massive amounts of unlabeled data. However, unsupervised SAE modeling does not guarantee to learn relevant representations for soft-sensing purposes. Therefore, the RB-PSOSAE aims to highlight the most significant features, retain the relevant ones, and remove the irrelevant ones. The obtained results demonstrated that RB-PSOSAE improved prediction performance compared to traditional search methods and meta-heuristics that only consider information from the MSE and do not deal with the dynamics of the process. Furthermore, RB-PSOSAE was more reliable and robust, as demonstrated in the same case study and under the same conditions. However, despite the contributions presented, future work could analyze the performance of the proposed method in other case studies, besides investigating the implementation of different metaheuristics and improving the evaluation function used.

Author Contributions: Conceptualization, A.G.V.S., J.M.M.d.L. and F.M.U.d.A.; methodology, A.G.V.S. and J.M.M.d.L.; software, A.G.V.S. and J.M.M.d.L.; validation, A.G.V.S., J.M.M.d.L. and F.M.U.d.A.; formal analysis, A.G.V.S., J.M.M.d.L. and F.M.U.d.A.; investigation, A.G.V.S. and J.M.M.d.L.; resources, A.G.V.S., J.M.M.d.L. and F.M.U.d.A.; data curation, A.G.V.S. and J.M.M.d.L.; writing—original draft preparation, A.G.V.S. and J.M.M.d.L.; writing—review and editing, A.G.V.S., J.M.M.d.L. and F.M.U.d.A.; visualization, A.G.V.S., J.M.M.d.L. and F.M.U.d.A.; supervision, A.G.V.S., J.M.M.d.L. and F.M.U.d.A.; project administration, A.G.V.S., J.M.M.d.L. and F.M.U.d.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Original debutanizer column and SRU datasets are available http://www.springer.com/cda/content/document/cda_downloaddocument/9781846284793_material.zip?SGWID=0-0-45-349600-p168288081 (accessed on 3 January 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Souza, F.A.A.; Araújo, R.; Mendes, J. Review of soft sensor methods for regression applications. *Chemom. Intell. Lab. Syst.* **2015**, *152*, 69–79. [CrossRef]
2. Zhong, W.; Yu, J. MIMO Soft Sensors for Estimating Product Quality with On-line Correction. *Chem. Eng. Res. Des.* **2000**, *78*, 612–620. [CrossRef]
3. Yuan, X.; Ge, Z.; Song, Z. Locally Weighted Kernel Principal Component Regression Model for Soft Sensing of Nonlinear Time-Variant Processes. *Ind. Eng. Chem. Res.* **2014**, *53*, 13736–13749. [CrossRef]
4. Ge, Z.; Huang, B.; Song, Z. Mixture semisupervised principal component regression model and soft sensor application. *Process. Syst. Eng.* **2014**, *60*, 533–545. [CrossRef]
5. Yang, X.; Liu, X.; Xu, C. Robust Mixture Probabilistic Partial Least Squares Model for Soft Sensing with Multivariate Laplace Distribution. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–9. [CrossRef]
6. Bao, L.; Yuan, X.; Ge, Z. Co-training partial least squares model for semi-supervised soft sensor development. *Chemom. Intell. Lab. Syst.* **2015**, *147*, 75–85. [CrossRef]
7. Yan, W.; Shao, H.; Wang, X. Soft sensing modeling based on support vector machine and Bayesian model selection. *Comput. Chem. Eng.* **2004**, *28*, 1489–1498. [CrossRef]
8. Shang, C.; Gao, X.; Yang, F.; Huang, D. Novel Bayesian Framework for Dynamic Soft Sensor Based on Support Vector Machine with Finite Impulse Response. *IEEE Trans. Control. Syst. Technol.* **2014**, *22*, 1550–1557. [CrossRef]
9. Yang, K.; Jin, H.; Chen, X.; Dai, J.; Wang, L.; Zhang, D. Soft sensor development for online quality prediction of industrial batch rubber mixing process using ensemble just-in-time Gaussian process regression models. *Chemom. Intell. Lab. Syst.* **2016**, *155*, 170–182. [CrossRef]
10. Li, X.; Su, H.; Chu, J. Multiple Model Soft Sensor Based on Affinity Propagation, Gaussian Process and Bayesian Committee Machine. *Chin. J. Chem. Eng.* **2009**, *17*, 95–99. [CrossRef]
11. Wang, G.; Jia, Q.S.; Zhou, M.; Bi, J.; Qiao, J.; Abusorrah, A. Artificial neural networks for water quality soft-sensing in wastewater treatment: A review. *Artif. Intell. Rev.* **2022**, *55*, 565–587. [CrossRef]
12. Pani, A.K.; Amin, K.G.; Mohanta, H.K. Soft sensing of product quality in the debutanizer column with principal component analysis and feed-forward artificial neural network. *Alex. Eng. J.* **2016**, *55*, 1667–1674. [CrossRef]
13. de Lima, J.M.M.; de Araújo, F.M.U. Industrial Semi-Supervised Dynamic Soft-Sensor Modeling Approach Based on Deep Relevant Representation Learning. *Sensors* **2021**, *21*, 3430. [CrossRef]
14. Patterson, J.; Gibson, A. *Deep Learning: A Practitioner's Approach*, 1st ed.; O'Reilly: Sebastopol, CA, USA, 2017; p. 214.
15. Weston, J.; Ratle, F.; Mobahi, H.; Collobert, R. Deep Learning via Semi-supervised Embedding. In *Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science*; Montavon, G., Orr, G.B., Müller, K.R., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7700._34 [CrossRef]
16. Zabalza, J.; Ren, J.; Zheng, J.; Zhao, H.; Qing, C.; Yang, Z.; Marshall, S. Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging. *Neurocomputing* **2016**, *185*, 1–10. [CrossRef]
17. Zhou, P.; Han, J.; Cheng, G.; Zhang, B. Learning Compact and Discriminative Stacked Autoencoder for Hyperspectral Image Classification. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 4823–4833. [CrossRef]
18. Qi, Y.; Wang, Y.; Zheng, X.; Wu, Z. Robust feature learning by stacked autoencoder with maximum correntropy criterion. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Florence, Italy, 4–9 May 2014; pp. 6716–6720. [CrossRef]
19. Wang, L.; You, Z.-H.; Chen, X.; Xia, S.X.; Liu, F.; Yan, X.; Song, K.J. A Computational-Based Method for Predicting Drug–Target Interactions by Using Stacked Autoencoder Deep Neural Network. *J. Comput. Biol.* **2018**, *25*, 361–373. [CrossRef] [PubMed]
20. Li, W.; Fu, H.; Yu, L.; Gong, P.; Feng, D.; Li, C.; Clinton, N. Stacked Autoencoder-based deep learning for remote-sensing image classification: A case study of African land-cover mapping. *Int. J. Remote Sens.* **2016**, *37*, 5632–5646. [CrossRef]
21. Blume, S.; Benedens, T.; Schramm, D. Hyperparameter Optimization Techniques for Designing Software Sensors Based on Artificial Neural Networks. *Sensors* **2021**, *21*, 8435. [CrossRef]
22. Kolar, D.; Lisjak, D.; Pająk, M.; Gudlin, M. Intelligent Fault Diagnosis of Rotary Machinery by Convolutional Neural Network with Automatic Hyper-Parameters Tuning Using Bayesian Optimization. *Sensors* **2021**, *21*, 2411. [CrossRef]
23. Aquino-Brítez, D.; Ortiz, A.; Ortega, J.; León, J.; Formoso, M.; Gan, J.Q.; Escobar, J.J. Optimization of Deep Architectures for EEG Signal Classification: An AutoML Approach Using Evolutionary Algorithms. *Sensors* **2021**, *21*, 2096. [CrossRef]
24. Kim, S.-H.; Geem, Z.W.; Han, G.-T. Hyperparameter Optimization Method Based on Harmony Search Algorithm to Improve Performance of 1D CNN Human Respiration Pattern Recognition System. *Sensors* **2020**, *20*, 3697. [CrossRef]
25. Brodzicki, A.; Piekarski, M.; Jaworek-Korjakowska, J. The Whale Optimization Algorithm Approach for Deep Neural Networks. *Sensors* **2021**, *21*, 8003. [CrossRef] [PubMed]
26. Weiming, S.; Xuemin, T.; Ping, W.; Xiaogang, D.; Sheng, C. Online soft sensor design using local partial least squares models with adaptive process state partition. *Chemom. Intell. Lab. Syst.* **2015**, *114*, 108–121. ISSN 0169-7439. j.chemolab.2015.04.003. [CrossRef]
27. Bank, D.; Koenigstein, N.; Giryes, R. Autoencoders. *arXiv* **2020**. [CrossRef]
28. Chadha, G.S.; Rabbani, A.; Schwung, A. Comparison of semi-supervised deep neural networks for anomaly detection in industrial processes. In Proceedings of the 2019 IEEE 17th International Conference on Industrial Informatics (INDIN), Helsinki-Espoo, Finland, 22–25 July 2019; Volume 1, pp. 214–219.

29. Bengio, Y.; Lamblin, P.; Popovici, D.; Larochelle, H. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2006; p. 19.
30. Hinton, G.E.; Salakhutdinov, R.R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504–507. [CrossRef] [PubMed]
31. Yan, X.; Wang, J.; Jiang, Q. Deep relevant representation learning for soft sensing. *Inf. Sci.* **2020**, *514*, 263–274. [CrossRef]
32. Eberhart, R.C.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the Sixth International Symposium on Micro Machine and Human Science, New York, NY, USA, 4–6 October 1995; pp. 39–43.
33. Elbes, M.; Alzubi, S.; Kanan, T.; Al-Fuqaha, A.; Hawashin, B. A survey on particle swarm optimization with emphasis on engineering and network applications. *Evol. Intel.* **2019**, *12*, 113–129. [CrossRef]
34. Singh, P.; Chaudhury, S.; Panigrahi, B.K. Hybrid MPSO-CNN: Multi-level Particle Swarm optimized hyperparameters of Convolutional Neural Network. *Swarm Evol. Comput.* **2021**, *63*, 100863. [CrossRef]
35. Sun, Y.; Xue, B.; Zhang, M.; Yen, G.G. An Experimental Study on Hyper-parameter Optimization for Stacked Auto-Encoders. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8. [CrossRef]
36. Haidong, S.; Ziyang, D.; Junsheng, C.; Hongkai, J. Intelligent fault diagnosis among different rotating machines using novel stacked transfer auto-encoder optimized by PSO. *ISA Trans.* **2020**, *105*, 308–319. [CrossRef]
37. Eberhart, R.C.; Shi, Y. Particle swarm optimization: Developments, applications and resources. In Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, Korea, 27–30 May 2001; pp. 81–86. [CrossRef]
38. Goodfellow, I.; Yoshua, B.; Aaron, C. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
39. Xiaochen, S.; Junxia, M.; Weili, X. Smart Soft Sensor Design with Hierarchical Sampling Strategy of Ensemble Gaussian Process Regression for Fermentation Processes. *Sensors* **2020**, *20*, 1957. [CrossRef]
40. Cofta, P.; Kostas, K.; and Cezary, O. A conceptual model of measurement uncertainty in iot sensor networks. *Sensors* **2021**, *21*, 1827. [CrossRef] [PubMed]
41. Yuan, X.; Wang, Y.; Yang, C.; Gui, W. Stacked isomorphic autoencoder based soft analyzer and its application to sulfur recovery unit. *Inf. Sci.* **2020**, *534*, 72–84. [CrossRef]
42. Pan, B.; Jin, H.; Wang, L.; Qian, B.; Chen, X.; Huang, S.; Li, J. Just-in-time learning based soft sensor with variable selection and weighting optimized by evolutionary optimization for quality prediction of nonlinear processes. *Chem. Eng. Res. Des.* **2019**, *144*, 285–299. [CrossRef]
43. Fortuna, L.; Graziani, S.; Rizzo, A.; Xibilia, M.G. *Soft Sensors for Monitoring and Control of Industrial Processes*; Springer: London, UK, 2007; Volume 22. [CrossRef]
44. Zhu, J.; Ge, Z.; Song, Z. Robust semi-supervised mixture probabilistic principal component regression model development and application to soft sensors. *J. Process Control* **2015**, *32*, 25–37. [CrossRef]
45. Fortuna, L.; Rizzo, A.; Sinatra, M.; Xibilia, M.G. Soft analysers for a sulfur recovery unit. *Control Eng. Pract.* **2003**, *11*, 1491–1500. [CrossRef]