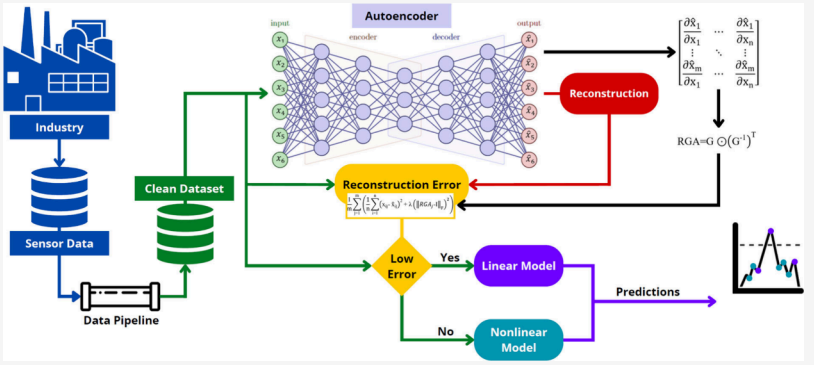
I&EC research
Industrial & Engineering Chemistry Research

pubs.acs.org/IECR
Article

Enhancing Autoencoder-Based Machine Learning through the Use of Process Control Gain and Relative Gain Arrays as Cost Functions

Rafael H. Martello,* Jorge O. Trierweiler, Lucas Maciel, and Marcelo Farenzena

Cite This: *Ind. Eng. Chem. Res.* 2024, 63, 16814−16822
Read Online



ABSTRACT: Autoencoders are neural networks utilized for unsupervised learning and reconstructing input data, making them helpful in for analyzing industrial process data. To enhance their effectiveness, we introduce two cost functions based on the Gain Matrix and Relative Gain Array (RGA) concepts, referred to in this paper as Gain Autoencoder (GAE) and Relative Gain Autoencoder (RGAE). These cost functions aid in reducing dimensionality and improving the model's performance in industrial settings. This article delves into applying of these functions in machine learning, particularly in autoencoders, to predict Mooney viscosity in styrene butadiene rubber (SBR) production. The findings indicate that the proposed GAE and RGAE models outperform traditional linear (linear regression) and nonlinear models (SVR), as evidenced by an increased explained variance of up to 10% and a decrease in mean square error of up to 13%. The successful integration of advanced data analysis techniques with domain knowledge in process control systems opens up new avenues for optimizing industrial processes and resource utilization.

1. INTRODUCTION

The exponential growth of generated data across diverse knowledge domains has led to a time where predictive models are crucial to create valuable insights and assist decision-making.[1] Among these predictive techniques, machine learning models have become popular in engineering fields because they can perform tasks previously limited to humans. As a result, industries, commerce, and financial markets are adopting these models to aid decision-making. Artificial neural networks, in particular, have successfully addressed challenges such as pattern classification, regression problems, and image identification, among others.[2,3]

Neural networks are computational models inspired by the structure of the human brain.[4] They comprise interconnected nodes, or neurons, organized into layers that process data through mathematical operations and activation functions, updating their weights through an optimization process. This enables them to undertake tasks such as image recognition, natural language processing, and decision-making with high accuracy. During training, neural networks employ cost functions, also known as loss functions, to quantify the disparity between predicted outputs and actual targets. By minimizing the cost function value using optimization algorithms like gradient descent, the neural network fine-tunes its parameters.[5]

Autoencoders are a type of neural network that learns without supervision. They were first introduced as models trained to reconstruct their input data by understanding the relationships between input variables rather than using labeled data.[6,7] The interaction between neurons within an autoencoder is an important aspect of this neural network design. Through training, the autoencoder aims to minimize the reconstruction error between the input and output layers, thereby acquiring the ability to reproduce input data from its

Received: January 30, 2024
Revised: September 1, 2024
Accepted: September 3, 2024
Published: September 23, 2024

https://doi.org/10.1021/acs.iecr.4c00343
Ind. Eng. Chem. Res. 2024, 63, 16814−16822

encoded representation.[8] The dimensionality reduction of the encoded space allows for efficient representation and data compression, capturing the critical features of the input data.[9]

Given this ability to learn meaningful data representations, autoencoders find extensive applications across diverse fields. In data compression tasks, they efficiently reduce the dimensionality of complex data sets while preserving essential information. In anomaly detection, autoencoders excel at recognizing deviations from standard patterns by assessing reconstruction errors, making them valuable tools for predictive maintenance, quality control, and process optimization. Furthermore, autoencoders play a significant role in image denoising, feature extraction, and data augmentation within computer vision applications.[10]

In the context of industrial processes, the autoencoder's ability to efficiently manage large data sets is highly advantageous. In complex manufacturing settings, autoencoders process a plethora of sensor data efficiently, enabling real-time monitoring and control of various processes.[11] Their ability to leverage auxiliary variables readily available in industrial settings further enhances their performance and interpretability.[7,12,13]

While the advantages of autoencoders are evident, it is important to recognize their limitations. One challenge lies in the potential for chaos within the latent space, where the autoencoder may struggle to generate organized structures due to nonspecific cost functions. This can hinder the achievement of meaningful representations and patterns. To mitigate this issue, careful regularization and validation techniques become imperative to ensure that the latent space captures relevant and interpretable features. Additionally, selecting an appropriate architecture and hyperparameters can prove complex and time-consuming, necessitating thorough experimentation to achieve optimal performance.[14]

Like the process industry, interaction between process channels can lead to control problems. In multivariable processes, appropriate output and input variables pairing can reduce the channel interactions.[15] The Relative Gain Array (RGA) concept, introduced by Bristol in 1966 for industrial control problems, quantifies interaction and applies to multi-input, multioutput scenarios. RGA analysis is commonly employed to identify decentralized multiloop control systems, especially when information is limited and steady-state gains are available. It significantly determines critical closed-loop system properties such as stability, robustness, decentralized integral controllability, and fault tolerance.[16]

This article thoroughly examines the application of machine learning models, specifically autoencoders, in predicting Mooney viscosity using data from a styrene butadiene rubber (SBR) production plant. The "Methodology" section provides an overview of autoencoders, discusses their potential differences, and introduces the Multivariate Process Control Gain and Relative Gain Array (RGA) as a possible solution applied in cost functions for training. It also covers data preprocessing, including shifts and feature engineering, and outlines the workflow and development process for model creation and optimization. We introduce the GAE and RGAE cost functions, which improve model interpretability and reconstruction quality. Using these functions, we examine their effectiveness in identifying linear and nonlinear data behaviors through a case study of actual data from an SBR production plant. The "Results and Discussion" section compares the performance of various models, including linear and nonlinear estimators. Our proposed approach significantly enhances the quality of autoencoder models, particularly for industrial applications, providing insights and solutions for optimizing rubber production processes and beyond.

2. METHODOLOGY

2.1. Multivariate Process Gain (G) and Relative Gain Array (RGA). In this section, we introduce the concepts of Multivariate Process Gain (G) and Relative Gain Array (RGA), examining how they could be integrated as cost functions in the training of autoencoders.

Gain is a fundamental concept in process control engineering, quantifying the relative impact of individual input variables on specific output variables within a multivariable system. It offers valuable insights into the dynamics of intricate processes by measuring how changes in input variables affect specific outputs while holding other inputs steady. This knowledge is essential for pinpointing key variables influencing system behavior significantly and devising effective control strategies.[17]

A square matrix of derivatives represents the Gain (G) according to

$$G = \begin{bmatrix} \frac{\partial \hat{x}_1}{\partial x_1} & \cdots & \frac{\partial \hat{x}_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \hat{x}_n}{\partial x_1} & \cdots & \frac{\partial \hat{x}_n}{\partial x_n} \end{bmatrix} \tag{1}$$

where

- $G \in \mathbb{R}^{nxn}$ is the gain matrix
- $x = [x_1, x_2, ..., x_n] \in \mathbb{R}$ is the input value of variables 1 to n
- $\hat{x} = [\hat{x}_1, \hat{x}_2, ..., \hat{x}_n] \in \mathbb{R}$ is the reconstructed outputvariables 1 to n
- $n \in \mathbb{N}^*$ is the number of variables

To compute the Gain matrix, a numerical method consists of applying small, finite perturbations to each input variable x_n while keeping the other inputs constant and then measuring the corresponding changes in the output variables $\hat{x}_n$. Specifically, each element of the Gain matrix can be approximated by the ratio of the change in an output variable to the change in the input variable, expressed as eq 2:

$$G \approx \begin{bmatrix} \frac{\Delta \hat{x}_1}{\Delta x_1} & \cdots & \frac{\Delta \hat{x}_1}{\Delta x_n} \\ \vdots & \ddots & \vdots \\ \frac{\Delta \hat{x}_n}{\Delta x_1} & \cdots & \frac{\Delta \hat{x}_n}{\Delta x_n} \end{bmatrix} \tag{2}$$

This finite difference approach provides a numerical approximation of the partial derivatives that constitute the Gain matrix. It is particularly beneficial in scenarios where an explicit mathematical model of the system is unavailable, allowing for the Gain matrix to be derived directly from experimental data. By systematically perturbing each input and measuring the resulting output changes, the Gain matrix offers a detailed representation of the system's sensitivity, which is essential for effective multivariable control strategies.

https://doi.org/10.1021/acs.iecr.4c00343
Ind. Eng. Chem. Res. 2024, 63, 16814−16822

Figure 1 shows a typical autoencoder architecture, which can depict the path from an input to an output using eq 3 for the case of n layers:

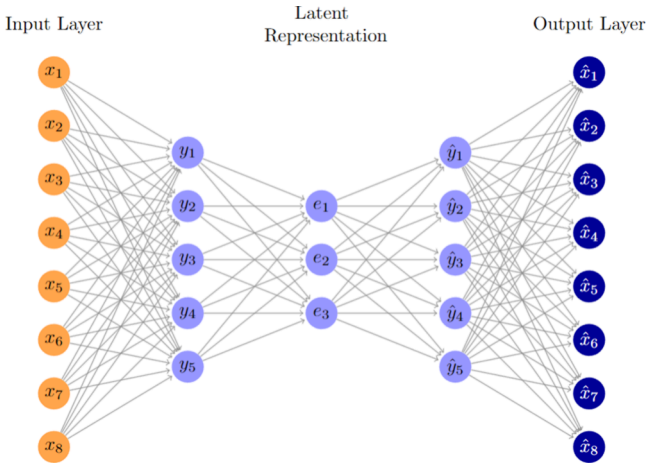


Figure 1. Autoencoder architecture.

$$\hat{x}_i = f_n\left(\sum_{i=1}^{m} W_{\hat{y}_i\hat{x}_j} f_{n-1}\left(\sum_{i=1}^{m} W_{e_{\hat{y}_j}} f_{n-2}\left(\ldots f_2\left(\sum_{i=1}^{m} W_{y_i e_j} f_1\left(\sum_{i=1}^{m} W_{x_{y_j}} x_i + b_1\right) + b_2\right) + \ldots\right) + b_{n-1}\right) + b_n\right) \tag{3}$$

where:

- $W_{x_{y_j}} \in \mathbb{R}$ are the weights connecting input i from layer $e = [x_i, y_i, e_i, \hat{y}_i]$ to neuron j from layer $s = [y_j, e_j, \hat{y}_j, \hat{x}_j]$
- $b = [b_1, b_2, \ldots, b_n] \in \mathbb{R}$ is the bias term for layer 1 to n
- $f: \mathbb{R} \rightarrow \mathbb{R}$ is the activation function applied from layer 1 to n

The derivatives of some activation functions can be ill-conditioned or undefined in certain regions. For example, the derivative of the ReLU function is undefined at zero, and the sigmoid function's derivative can approach zero for large positive or negative inputs, leading to numerical issues. To address this issue, it is possible to assign a value of zero in regions where the derivative is undefined. This treatment is based on the idea that, at these specific points, the contribution to gradient-based optimization is minimal or irrelevant. By setting the derivative to zero, we effectively neutralize any potential numerical instability that could arise from attempting to compute or propagate an undefined derivative. This ensures the stability and convergence of the optimization process, particularly in deep networks where such issues might accumulate and propagate through layers.

Examining the structure of Gain allows drawing parallels with an autoencoder. Similar to how an autoencoder maps input data into a lower-dimensional latent space and then reconstructs it in the output layer, the Gain matrix represents the impact of individual inputs on each output within a multivariable process.

Likewise, the Relative Gain Array (RGA) (4), introduced by Bristol in 1966, provides a matrix describing the relationships between input-output pairs in a control system. RGA assesses how alterations in each input variable affect each output variable while keeping the other inputs unchanged. This matrix reveals the connections and interactions among various variables, assisting engineers and researchers in identifying robust and weak interactions within the process. RGA consistently yields a matrix, where the sums of rows and columns equal 1. Its calculation can be expressed by the following equation:[17]

$$\mathrm{RGA} = G \odot (G^{-1})^T \tag{4}$$

where:

- $RGA \in \mathbb{R}^{nxn}$ is the resulting Relative Gain Array matrix,
- $\odot$ is the elementwise Hadamard product of the gain matrix (G) and $(G^{-1})^T$ the inverse transpose gain matrix

To leverage the potential of Gain and RGA, we suggest incorporating them into cost functions during the training of autoencoders. Through such incorporation, we aim to to augment the autoencoder's capacity to comprehend and depict intricate relationships among variables. Here, we are tuning the weights that produce the minimal interaction between the inputs and outputs. In other words, we are looking to create a network that will be as close as possible to one where an input change will mainly change the corresponding output, producing as small as possible change in the other outputs. Through this integration, autoencoders can effectively utilize the insights provided by Gain and RGA to guide the learning process. This leads to enhanced data representation, further empowering the autoencoder for various applications. By incorporating Gain principles into the learning process, the autoencoder can focus on capturing the influential interactions between process variables, thus generating a more informative and interpretable latent space. The RGA, derived from the Gain matrix, further complements the training process by identifying variable interactions that may lead to undesirable system behaviors. This aids in regularizing the latent space and promoting stable and robust representations, making the matrix as diagonal as possible.

2.2. Cost Functions. Generative models like autoencoders typically require a cost function to regulate the connection between input data and its reconstructed version. The commonly used cost function is MSE (mean squared error). However, these constraints may be overly simplistic. While MSE is easy to handle, it tends to prioritize outliers in the data when present. This phenomenon occurs because MSE assigns higher importance to outliers, thus diverting the model's optimization efforts toward these atypical points.[14]

The mean square error (5) was used as a base equation for the new cost functions.

$$\mathrm{MSE} = \frac{1}{m}\sum_{j=1}^{m}\left(\frac{1}{n}\sum_{i=1}^{n}(x_{ij}-\hat{x}_{ij})^2\right) \tag{5}$$

where:

- MSE $\in \mathbb{R}$ is the mean squared error, a non-negative real number
- $x \in \mathbb{R}$ is the input value of variables 1 to n, for the batch 1 to m
- $\hat{x} \in \mathbb{R}$ is the reconstructed outputvariables 1 to n, for the batch 1 to m
- $m \in \mathbb{N}^*$ the number of samples of the batch

https://doi.org/10.1021/acs.iecr.4c00343
Ind. Eng. Chem. Res. 2024, 63, 16814−16822

As previously introduced in our study,[18] the methodology of integrating the gain matrix and RGA into the cost function has the potential to improve the quality of unsupervised models. In this work, we enhanced the functions introduced in the previous study by subtracting an identity matrix. This addition is essential because a matrix identity can lead to a fully decoupled system in a nonreduced and entirely linear network, where each input exclusively influences a single output. By subtracting the identity matrix, we penalize the values different from the identity target matrix. Therefore, the first proposed cost function based on the Gain matrix (G) combined with mean square error (6).

$$f_{\text{gain}} = \frac{1}{m}\sum_{j=1}^{m}\left(\frac{1}{n}\sum_{i=1}^{n}(x_{ij}\text{-}\hat{x}_{ij})^2 + \lambda(\| G_j - I \|_{\text{F}})^2\right) \tag{6}$$

where:

- $f_{\text{gain}} \in \mathbb{R}$ is the resulting cost function value.
- $I \in \mathbb{R}^{nxn}$ is an identity matrix with the same dimensions as G,
- $\lambda \in \mathbb{R}$ is the relative importance of the gain contribution.

Based on widely studied regularization techniques like Lasso (L1) and Ridge (L2) as described in Melkumova and Shatskikh (2017),[19] the function was developed employing a lambda (λ) to define the relative importance of the gain matrix. The regularization value comprises the squared Frobenius norm of the difference between the gain matrix and an identity matrix of equivalent size (7), whose function is to correct the value to an ideal system (with no interaction between channels and all layers of the same size). The Frobenius norm is givem by

$$\| G_j - I \|_{\text{F}} = \sqrt{\sum_{i=1}^{n_y}\sum_{j=1}^{n_x}(g_{ij} - \delta_{ij})^2} \tag{7}$$

where:

- $g_{ij} \in \mathbb{R}$ are the elements of the matrix G
- $\delta_{ij} \in \mathbb{R}$ are the elements of the matrix I,
- Each element δ_{ij} satisfies

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

- n_x and $n_y \in \mathbb{N}^*$ are the dimensions of matrix G and I, with $n_x = n_y$ for autoencoders

The Frobenius norm is a matrix norm that extends the concept of the Euclidean norm to matrices. It is the square root of the sum of the absolute squares of a matrix's elements. The Frobenius norm is particularly useful because it is invariant under orthogonal transformations, meaning that for any orthogonal matrices Q and P, $\|QAP\|_{\text{F}} = \|A\|_{\text{F}}$. This property makes the Frobenius norm popular in various applications, including matrix approximations, machine learning algorithms, and numerical linear algebra.

Another significant feature of the Frobenius norm is its equivalence to the Euclidean norm when the matrix is treated as a vector in $\mathbb{R}^{nxn}$. This equivalence provides an intuitive geometric interpretation and aligns the Frobenius norm with familiar norms from vector space theory.

In the context of optimization problems and regularization techniques, the Frobenius norm is frequently used to measure the size or complexity of a matrix, aiding in the prevention of overfitting by penalizing large matrices.

Using of the Frobenius norm to penalize the difference between matrices in an autoencoder is motivated by highlighting and amplifying significant individual errors. Such a choice is made to assign of assigning greater importance to notable deviations between the actual and ideal coupling values,which leads to a selective penalization strategy. By selecting the Euclidean norm, the emphasis is placed on a more localized assessment of errors, enabling a more targeted approach to address specific coupling patterns within the autoencoder architecture.

Likewise, the following cost function $f_{\text{RGA}} \in \mathbb{R}$ was developed for the RGA, which is given by (8):

$$f_{\text{RGA}} = \frac{1}{m}\sum_{j=1}^{m}\left(\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \hat{x}_{ij})^2 + \lambda(\| RGA_j - I \|_{\text{F}})^2\right) \tag{8}$$

i.e., the RGA was used in the cost function, resembling eq 6, with a lambda value (λ) and the squared Frobenius norm of the difference between RGA and an identity matrix.

2.3. Virtual Analyzer Methodology. The methodology employed herein consists of a comprehensive data-driven approach using advanced process control and machine learning techniques to develop a virtual analyzer. Data preparation and analysis were done using Python v. 3.9.16, leveraging specific libraries tailored for various tasks. Pandas v. 2.0.1 and NumPy v.1.21.5 were utilized for data manipulation and feature engineering, including the implementation of necessary shifts for optimal time synchrony analysis. PyTorch v.2.0.0 and Scikit-learn v.1.2.2 were used for creating and training predictive models, leveraging their capabilities for modeling and optimizing autoencoder-based predictions.

2.4. Data Processing. Preprocessing played a crucial role in preparing the data for training and validating the models using actual process unit data as outlined in the case study. This involved several feature engineering steps, including assessing temporal synchronization (synchrony) between variables, expanding time series, evaluating variable averages, and data cleansing to ensure its quality and reliability.

Temporal synchronization, often called to as synchrony, represents the alignment or coordination of events or changes over time. In the context of data analysis, particularly in industrial processes, synchrony refers to the degree to which different variables exhibit simultaneous or correlated patterns of variation over time. It indicates the temporal relationship between various process parameters, reflecting how changes in one variable correspond to changes in another within a specific time frame. Achieving synchrony in data analysis involves ensuring that the temporal aspects of different variables are appropriately accounted for, allowing for a more accurate understanding of their interactions and dynamics over time. This synchronization is crucial for identifying causal relationships, detecting anomalies, and making reliable predictions in time-series data analysis.

Given the process's dynamics, some variables do not synchronize instantly with the variable of interest. Instead, their impact becomes apparent only after a specific time interval, reflecting the temporal delay associated with the product's progression through different process stages. This

https://doi.org/10.1021/acs.iecr.4c00343
Ind. Eng. Chem. Res. 2024, 63, 16814−16822

delay is quantified by the residence time, representing the duration necessary for the product to traverse the equipment and arrive at the collection point.

Achieving an optimal temporal alignment between secondary and target variable required data processing. We implemented a data analysis methodology based on temporal lag estimation in this analysis. This methodology involved leveraging algorithms capable of maximizing the correlation between variables across different lag intervals. A statistical technique to systematically identify the optimal lag for each variable was employed using Pearson's correlation.

Pearson correlation, a widely used statistical measure, quantifies the strength and direction of the linear relationship between two variables. It assesses the degree to which changes in one variable are associated with changes in another, providing insights into their linear dependence. Pearson correlation is valuable for identifying interdependencies between different process parameters in industrial processes, indicating how closely their variations align over time. When applied to time-series data, Pearson correlation helps reveal temporal associations between variables, highlighting patterns of synchronization or lagged effects.

By doing so, we ensured a comprehensive exploration of potential lag effects, considering the dynamic nature of the polymerization process. This approach enabled us to capture the intricate relationships and time dependencies among the variables more accurately, enhancing our predictive models' robustness. The correlations were plotted (as depicted in Figure 2). The plot explored increasing time offsets up to the maximum residence time feasible. The aim was to visually identify the time shift that produced the highest correlation, pinpointing the maximum synchronization point.

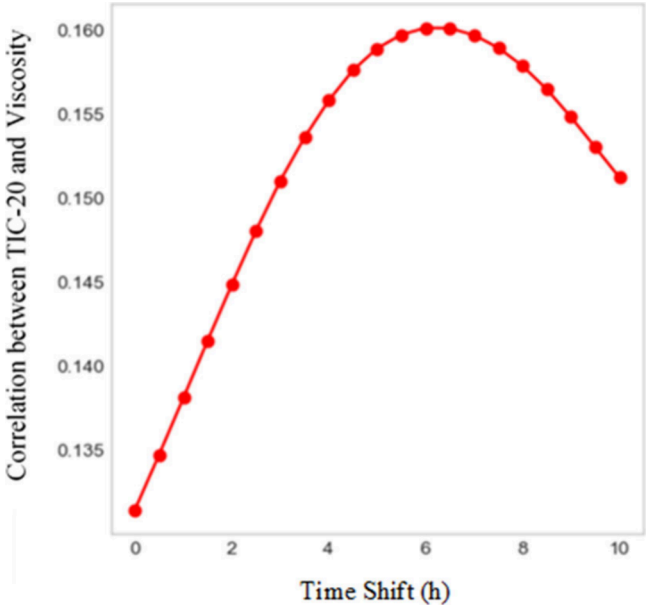


Figure 2. Optimal time synchrony for variable TIC-20.

After determining the optimal time lag between independent variables and the response variable, a feature engineering process was conducted to create new features that more accurately captured the underlying process while mitigating the impact of outliers. The case study further details these resulting variables, offering a more comprehensive understanding of their characteristics. Following the introduction of these new variables, the data set underwent normalization using a Standard Scaler. This rescaled the value distribution, ensuring the mean of observed values was centered at zero, and the standard deviation was set to 1. Due to the time series format of the data, the data set was partitioned in a 70:30 ratio based on the temporal variable for training and evaluation. This partitioning allowed for a thorough data set exploration, ensuring an effective balance between training and testing data for subsequent analyses.

2.5. Feature Selection. Feature selection using Lasso Lars (Least Angle Regression) regularization is an essential component of our methodology, aimed at identifying the most relevant variables for predictive modeling. Lasso Lars is a well-established technique known for simultaneously performing variable selection and regularization,, making it suitable for our purposes.

In our approach, Lasso Lars systematically evaluates each feature's contribution to the model's predictive performance. By penalizing the coefficients of less influential features, the algorithm effectively filters out variables with minimal impact on prediction accuracy. This iterative process allows us to refine the feature set, retaining only those variables that significantly contribute to the model's explanatory power.

During the training phase, the algorithm undergoes multiple iterations, adjusting the penalty term (alpha) to find the optimal balance between model complexity and predictive performance. The choice of alpha is important, as it controls the degree of regularization applied to the model. Higher alpha values lead to more aggressive regularization, resulting in simpler models with potentially lower predictive accuracy. Conversely, lower alpha values allow for more flexibility in the model but may lead to overfitting.

By carefully monitoring the coefficient of determination (R^2), we ensure that the selected features maximize prediction accuracy while maintaining model parsimony. The final set of features identified by Lasso Lars represents the most informative variables in our data set, contributing significantly to the model's predictive performance.

Overall, the feature selection process using Lasso Lars improves the efficiency and interpretability of our predictive models. We can develop more accurate and interpretable models by selecting a subset of features with the greatest predictive power, facilitating better decision-making in various applications.

The initial set, comprising 60 variables, was reduced to 20 variables. This reduction was achieved through iterative adjustments of the penalty term (alpha) within the model in an iterative process from 1×10^{-6} to 1 multiplying by 10 every step reaching an optimal value of 1×10^{-2}.

The process involved careful evaluation based on the coefficient of determination (R^2) for prediction accuracy. By iteratively tuning the penalty term, we aimed to balance the preserving model complexity and ensuring optimal predictive performance. The selected 20 features were identified as significant contributors to the model's overall explanatory ability, thereby enhancing the efficiency and interpretability of our predictive framework. This feature selection approach enhanced computational efficiency and facilitated a more focused analysis of variables with greater predictive power in the data set.

2.6. Anomaly Detection. Researchers commonly leverage significant volumes of routine data to develop anomaly

https://doi.org/10.1021/acs.iecr.4c00343
Ind. Eng. Chem. Res. 2024, 63, 16814−16822

detection methodologies. These approaches typically fall under semisupervised anomaly detection, where only standard process data is utilized to train reconstruction-based models. These models identifyanomalies by analyzing reconstruction errors, employing specific decision thresholds to distinguish between normal and abnormal behavior.[20]

Following the principles of semisupervised anomaly detection, we established a workflow depicted in Figure 3,

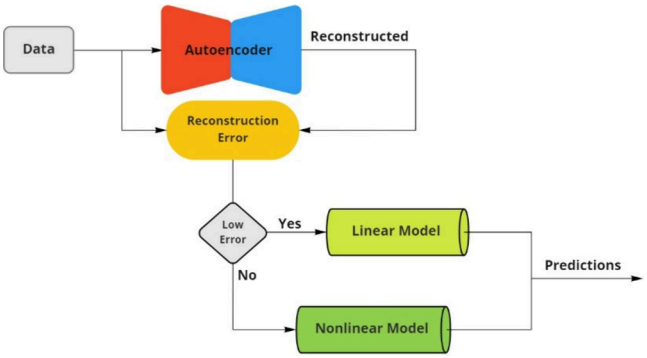


Figure 3. Workflow for model predictions.

utilizing an autoencoder. The autoencoder was trained on data exhibiting low residual reconstruction error, serving as a filter for observations. It effectively captures observations characterized by lower reconstruction error, indicative of linear behavior, which are subsequently used in a linear regression model. Conversely, Support Vector Regressor (SVR) nonlinear model evaluates observations with high reconstruction errors.

Autoencoder hyperparameters were determined through a Grid Search approach, exploring various activation functions (Tanh, ReLU, and Sigmoid) and learning rates (1×10^{-4}, 1×10^{-3}, and 1×10^{-2}). The chosen model comprised a deep autoencoder architecture featuring an intermediate layer of 8 neurons and an encoder with five neurons, utilizing Rectified Linear Units (ReLU) as the activation function, the Adam optimizer, and a learning rate of 1×10^{-3}. The model underwent evaluation over 100 epochs, initialized with the Xavier uniform function as described by Glorot and Bengio (2010).[21] The lambda value was determined through grid search, with the optimal value identified as 0.5.

Following the workflow outlined in Figure 3, two prediction models were developed for the case study. Initially, a linear regression model and a Support Vector Regressor (SVR) model were trained using the first data set (train and test) to serve as nonlinear estimators. The hyperparameters of these models underwent meticulous optimization through Grid Search, exploring various kernel options, including linear, polynomial, and Radial Basis Function (RBF). The SVR model with an RBF kernel, epsilon of 0.5, gamma of 0.001, and C of 10, was identified as the optimal configuration.

The training process for the autoencoder commenced with partitioning the data set into separate training and validation sets, with a portion allocated for validation purposes. Given the temporal nature of the data within an industrial setting, characterized by sensor measurements over time, the data set was segmented accordingly. Specifically, the training set comprised data spanning from December 2020 to July 2021, while the validation set encompassed one month from August 2021 to September 2021. A collaborating company graciously provided these data sets.

2.7. Evaluation and Metrics. A range of assessment metrics were utilized to evaluate the models' performance, including the Regression Coefficient (r^2) to gauge the model's quality, Mean Squared Error (MSE), and explained variance. Additionally, for the autoencoder-based models, the Mean Reconstruction Error (MRE) was also considered. These comprehensive metrics enabled a detailed examination of the models' capabilities and effectiveness in predicting the variable of interest in the case study.

Seeking a comparison with other models, a similar structure was also created using partial least-squares (PLS) with 5 components, used as an anomaly filter instead of the autoencoder and evaluated based on the same statistics. Partial Least Squares (PLS) is a multivariate regression technique commonly used in data analysis, particularly when dealing with high-dimensional data sets or collinear predictors. Unlike traditional least-squares regression, which builds a model directly on the original predictors, PLS constructs latent variables (also known as components) that capture the maximum covariance between the predictors and the response variable. These latent variables are iteratively calculated by extracting information from both the predictors and the response variable, making PLS particularly effective in when the number of predictors exceeds the number of observations or when there are strong correlations among predictors. PLS aims to find a linear relationship between the predictors and the response by maximizing their covariance, thus facilitating prediction and interpretation in complex data sets.

3. CASE STUDY

Data were gathered from sensors in an operational styrene−butadiene rubber (SBR) polymerization process within an industrial setting. These data were collected during two distinct periods, resulting in two data sets: one used for model training and testing, and the other for production validation. The initial data set (train/test) consisted of 57,302 observations from 42 sensors, spanning 221 process days. The second data set (production) comprised 7,875 observations spanning 31 process days.

This case study focuses on a plant that produces Styrene Butadiene Rubber (SBR) through cold polymerization involving free radicals. Mooney viscosity is a vital quality indicator for assessing the rubber's performance. It quantifies the elastomer's strength under shear forces within the elastic regime. This parameter is closely linked to the processing conditions required for producing valuable products, such as tires, through vulcanization. As Mooney viscosity significantly influences customer decisions when purchasing rubber for further processing, we aim is to develop a predictive model to estimate this critical property.

The polymerization reaction occurs in of 15 Continuous Stirred-Tank Reactors (CSTRs) in series, each equipped with temperature control systems utilizing tubular bundles containing liquid ammonia. These bundles are submerged in the reaction mixture. As the reaction progresses, the heat released is transferred to ammonia, causing it to vaporize and regulate the reactor temperature within a specified range of 8 to 12 °C. The reaction mixture, referred to as latex, continues until it reaches a 66% conversion rate. This specific rate ensures that approximately 34% of unreacted monomers (butadiene and styrene) remain, which are then removed in the Monomer Recovery area for reuse in the process.

https://doi.org/10.1021/acs.iecr.4c00343
Ind. Eng. Chem. Res. 2024, 63, 16814−16822

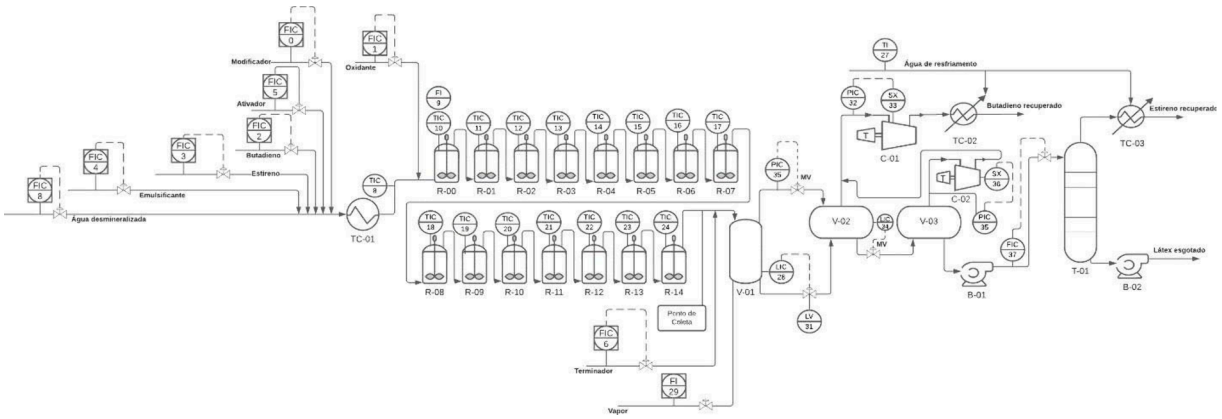


Figure 4. Process flowchart of the polymerization unit.

To effectively monitor and control the polymerization process, a total of 42 variables are collected via industrial sensors, as shown in Figure 4, following the process flowchart. These variables encompass the flow rates of components (FIC-0 to FI-9, FI-29, FIC-37, 39), reactor temperatures (TIC10 to TIC24 and TI-27), process pressures (PIC-25, PIC-30.MV, PIC-32, and PIC-35), compressor capacities (SX-26, SX-33, and SX-36), vessel level (LIC-28), valve openings (LV-31 and LIC-34.MV), the number of active reactors (38), the reaction rate (40), butadiene/styrene ratio (41), and laboratory-measured Mooney viscosity obtained every 4 h (42).

Additionally, variables related to reagent concentrations (52, 55, 56, 57, and 58), ammonia flow (59), and mean reactor temperatures (60, 61, 62, and 63) have been incorporated to enhance our understanding of the process further and improve Mooney viscosity estimation.

Three additional variables were incorporated into the data set, derived from the temperatures of the reactor chain: variable 45 represents the average temperature from R-00 to R-04, variable 46 represents the average temperature from R-05 to R-09, and variable 47 represents the average temperature from R-10 to R-15. These new variables were added to mitigate the adverse impact of reactors occasionally being taken out of operation on the temperature model. Such occurrences could lead to outliers caused by temperatures exceeding control limits, without significantly affecting the primary variable.

This case study aims to investigate the relationships among these process variables and develop a dependable inference model for estimating Mooney viscosity. By doing so, we strive to optimize the polymerization process, ensure consistent rubber quality, and improve the overall efficiency of the SBR rubber production plant.

4. RESULTS AND DISCUSSION

We initiated our analysis by meticulously selecting variables using the LASSO Lars model, a method known for its efficacy in simultaneous variable selection and regularization. This process yielded a total of 20 variables deemed significant for our predictive models. The selected variables and detailed descriptions are meticulously documented and presented in Table 1, providing valuable insights into the features contributing to the model's performance.

Subsequently, we evaluated both linear and nonlinear models, followed by their integration using the autoencoder workflow depicted in Figure 3. The reconstruction error served as a pivotal parameter for segregating observations. The comparative analysis of model results obtained from the production data set is presented in Table 2, offering insights into key metrics such as the coefficient of regression (r^2), Mean Squared Error (MSE), explained variance, mean reconstruction error, and computational time.

Table 1. Selected Features for the Model

Var	Description	Var	Description
FIC-0	Flow of modifier	55	Butadiene concentration
TIC23	Reactor 13 temperature	56	Emulsifier concentration
TIC24	Reactor 14 temperature	57	Activator concentration
PIC-25	Ammonia pressure	58	Concent. demineralized water
PIC-30.$_{MV}$	Vapor-phase valve in accumulation vessel	59	Mean ammonia flow
38	Number of active reactors	60	Mean reactor temperature 7
39	Polymerization flow	61	Mean reactor temperature 9
45	The average temperature of R-00−R-04	62	Mean reactor temperature 13
46	The average temperature of R-05−R-09	63	Mean reactor temperature 14
47	The average temperature of R-10−R-15	52	Modifier concentration
52	Modifier concentration		

A model using partial least-squares was implemented within the same anomaly analysis framework for comparison purposes, aiming to maintain comparability. As observed in the data shown in Table 2, the developed models took a few additional minutes to train but yielded better results, statistically distinguishing themselves from the other models. It is also worth noting that training time for real-time production line applications is not a concern, considering the model's potential to estimate on-site viscosity without laboratory analyses. We observed statistically significant differences in model performance by integrating newly devised cost functions, namely Gain Autoencoder (GAE) and Relative Gain Autoencoder (RGAE). These differences were corroborated by increased regression coefficients and reduced errors, as verified through statistical analyses including the Tukey test.

The outcomes of our methodology are further supported by graphical representations, such as Figure 5, which provides a visual depiction of the time-series predictions generated by the

https://doi.org/10.1021/acs.iecr.4c00343
Ind. Eng. Chem. Res. 2024, 63, 16814−16822

Table 2. Comparison of Model Results in Production Data

Model	Coefficient of Regression (r^2)	MSE	Explained Variance	Mean Reconstruction Error	Computational Time [minutes]
Linear	62.64%[a]	7.05[a]	64.10%[a]		1.12 ± 0.07
Nonlinear	61.68%[a]	7.22[a]	61.92%[a]		1.26 ± 0.03
PLS	28.68[b]	13.49[b]	62.68[b]	0.54	
AE	63.04 ± 1.22%[a]	7.33 ± 0.242[c]	63.72 ± 0.83%[a]	0.4223 ± 0.0698	4.47 ± 0.43
GAE	67.53 ± 2.16%[c]	6.44 ± 0.4278[c]	67.68 ± 2.17%[c]	0.3646 ± 0.0362	7.04 ± 2.13
RGAE	68.07 ± 2.43%[c]	6.33 ± 0.4819[c]	68.55 ± 2.47%[c]	0.4223 ± 0.0698	6.75 ± 3.11

[a]First group in Tukey Test. [b]Second group in Tukey Test. [c]Third group in HDS Tukey Test.

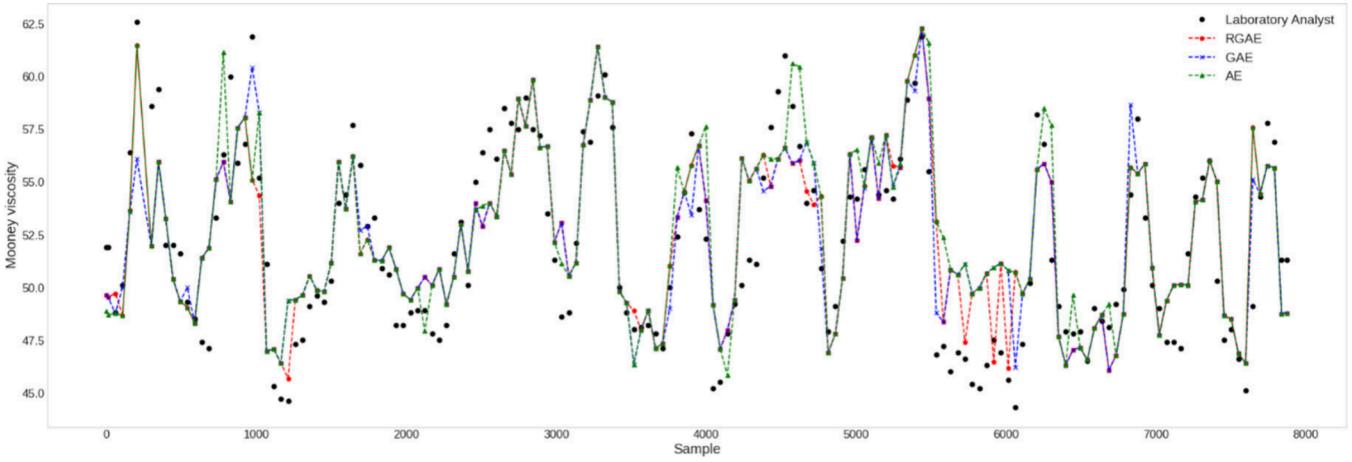


Figure 5. Time-series predictions.

combined models. Additionally, references to related studies underscore the broader applicability of similar methodologies, emphasizing the substantial potential of autoencoders in data analysis and decision-making processes.

Overall, our results underscore the efficacy of the developed virtual analyzer in process control applications. By maintaining error values within acceptable thresholds, our approach facilitates efficient monitoring of Mooney viscosity, leading to enhanced product quality, reduced storage duration, cost savings through decreased water vapor consumption, and improved uniformity in the molecular weight distribution of the final product's polymer chains. These findings hold significant implications for industries reliant on process optimization and quality control measures.

5. CONCLUSIONS

In conclusion, this study effectively tackled the challenge of accurately estimating Mooney viscosity for SBR production through a data-driven approach. Leveraging machine learning and autoencoder-based models resulted in statistically significant differences in predictive accuracy compared to conventional linear and nonlinear models. The integration of Gain Matrix and Relative Gain Array (RGA) concepts as cost functions played a crucial role in improving the model's performance, enabling better understanding and management of the polymerization process.

Our virtual analyzer emerges as a valuable resource for the industry, providing real-time process monitoring and quality control while reducing the need for frequent laboratory analyses. This refined monitoring approach not only boosts the overall efficiency of the production plant but also ensures consistent rubber quality and cost efficiency. The successful utilization of autoencoders in data behavior analysis further underscores their potential for diverse industrial applications, surpassing traditional anomaly detection methods.

By combining advanced data analysis techniques with domain knowledge of multivariable process control systems, this study paves the way for refining and optimizing polymerization processes across various industries.

AUTHOR INFORMATION

Corresponding Author

Rafael H. Martello − *Group of Intensification, Modeling, Simulation, Control, and Optimization of Process, GIMSCOP, Federal University of Rio Grande do Sul, Chemical Engineering Department, 90040-040 Porto Alegre, Rio Grande do Sul, Brazil;* orcid.org/0000-0003-2793-5087; Phone: +55 49-98830-5836; Email: rhmartello@gmail.com

Authors

Jorge O. Trierweiler − *Group of Intensification, Modeling, Simulation, Control, and Optimization of Process, GIMSCOP, Federal University of Rio Grande do Sul, Chemical Engineering Department, 90040-040 Porto Alegre, Rio Grande do Sul, Brazil;* orcid.org/0000-0002-6328-945X

Lucas Maciel − *Group of Intensification, Modeling, Simulation, Control, and Optimization of Process, GIMSCOP, Federal University of Rio Grande do Sul, Chemical Engineering Department, 90040-040 Porto Alegre, Rio Grande do Sul, Brazil*

Marcelo Farenzena − *Group of Intensification, Modeling, Simulation, Control, and Optimization of Process, GIMSCOP, Federal University of Rio Grande do Sul, Chemical Engineering Department, 90040-040 Porto Alegre, Rio Grande do Sul, Brazil*

https://doi.org/10.1021/acs.iecr.4c00343
Ind. Eng. Chem. Res. 2024, 63, 16814−16822

Complete contact information is available at:
https://pubs.acs.org/10.1021/acs.iecr.4c00343

Funding

The Article Processing Charge for the publication of this research was funded by the Coordination for the Improvement of Higher Education Personnel - CAPES (ROR identifier: 00x0ma614).

Notes

The authors declare no competing financial interest.

REFERENCES

(1) Ranzan, L.; Trierweiler, L. F.; Trierweiler, J. O. Prediction of Sulfur Content in Diesel Fuel Using Fluorescence Spectroscopy and a Hybrid Ant Colony - Tabu Search Algorithm with Polynomial Bases Expansion. *Chemometrics and Intelligent Laboratory Systems* **2020**, *206*, 104161.

(2) de Campos Souza, P. V. Fuzzy Neural Networks and Neuro-Fuzzy Networks: A Review the Main Techniques and Applications Used in the Literature. *Appl. Soft Comput* **2020**, *92*, No. 106275.

(3) Liu, X.; Tian, S.; Tao, F.; Yu, W. A Review of Artificial Neural Networks in the Constitutive Modeling of Composite Materials. *Compos B Eng.* **2021**, *224*, No. 109152.

(4) Gharbi, R. B. C.; Mansoori, G. A. An Introduction to Artificial Intelligence Applications in Petroleum Exploration and Production. *J. Pet Sci. Eng.* **2005**, *49* (3−4), 93−96.

(5) Zou, J.; Han, Y.; So, S. S. Overview of Artificial Neural Networks. *Methods Mol. Biol.* **2008**, *458*, 14−22.

(6) Bank, D.; Koenigstein, N.; Giryes, R. Autoencoders. *arXiv (Machine Learning)*, April 3, 2020, arXiv:2003.05991, v2. DOI: 10.48550/arXiv.2003.05991.

(7) Charte, D.; Charte, F.; García, S.; del Jesus, M. J.; Herrera, F. A Practical Tutorial on Autoencoders for Nonlinear Feature Fusion: Taxonomy, Models, Software and Guidelines. *Information Fusion* **2018**, *44*, 78−96.

(8) Chen, S.; Guo, W. Auto-Encoders in Deep Learning—A Review with New Perspectives. *Mathematics 2023, Vol. 11, Page 1777* **2023**, *11* (8), 1777.

(9) Trunz, E.; Weinmann, M.; Merzbach, S.; Klein, R. Efficient Structuring of the Latent Space for Controllable Data Reconstruction and Compression. *Graphics and Visual Computing* **2022**, *7*, No. 200059.

(10) Li, P.; Pei, Y.; Li, J. A Comprehensive Survey on Design and Application of Autoencoder in Deep Learning. *Appl. Soft Comput* **2023**, *138*, No. 110176.

(11) Alfeo, A. L.; Cimino, M. G. C. A.; Manco, G.; Ritacco, E.; Vaglini, G. Using an Autoencoder in the Design of an Anomaly Detector for Smart Manufacturing. *Pattern Recognit Lett.* **2020**, *136*, 272−278.

(12) Almotiri, J.; Elleithy, K.; Elleithy, A. Comparison of Autoencoder and Principal Component Analysis Followed by Neural Network for E-Learning Using Handwritten Recognition. *Proc. 2017 IEEE Long Island Syst., Appl. Technol. Conf.* **2017**, 1−5, DOI: 10.1109/LISAT.2017.8001963.

(13) Martinez-Murcia, F. J.; Ortiz, A.; Gorriz, J. M.; Ramirez, J.; Castillo-Barnes, D.; Salas-Gonzalez, D.; Segovia, F. Deep Convolutional Autoencoders vs PCA in a Highly-Unbalanced Parkinson's Disease Dataset: A DaTSCAN Study. *Advances in Intelligent Systems and Computing* **2019**, *771*, 47.

(14) Zhu, Q.; Wang, H.; Zhang, R. Wavelet Loss Function for Auto-Encoder. *IEEE Access* **2021**, *9*, 27101−27108.

(15) Salgado, M. E.; Conley, A. MIMO Interaction Measure and Controller Structure Selection. *Int. J. Control* **2004**, *77*, 367.

(16) Chen, D.; Seborg, D. E. Relative Gain Array Analysis for Uncertain Process Models. *AIChE J.* **2002**, *48* (2), 302−310.

(17) Skogestad, S.; Postlethwaite, I. *Multivariable Feedback Control: Analysis and Design*, 2nd ed.; John Wiley & Sons, Incorporated, 2005.

(18) Martello, R. H.; Ranzan, L.; Farenzena, M.; Trierweiler, J. O. Improving Autoencoder Training with Novel Goal Functions Based on Multivariable Control Concepts. *IFAC-PapersOnLine* **2021**, *54* (3), 73−78.

(19) Melkumova, L. E.; Shatskikh, S. Y. Comparing Ridge and LASSO Estimators for Data Analysis. *Procedia Eng.* **2017**, *201*, 746−755.

(20) Chow, J. K.; Su, Z.; Wu, J.; Tan, P. S.; Mao, X.; Wang, Y. H. Anomaly Detection of Defects on Concrete Structures with the Convolutional Autoencoder. *Advanced Engineering Informatics* **2020**, *45*, No. 101105.

(21) Glorot, X.; Bengio, Y. Understanding the Difficulty of Training Deep Feedforward Neural Networks. *PMLR* **2010**, *9*, 249−256.