A Pipeline for Multivariate Time Series Forecasting of Gas Consumption in Pelletization Process

Thadeu Pezzin Melo,
Jefferson Oliveira Andrade,
Karin Satie Komati
Instituto Federal do Espírito Santo, Campus Serra
Programa de Pós-graduação em Computação Aplicada
Serra, Brazil, 29166-630,
tpezzin@gmail.com
{*jefferson.andrade, kkomati*}*@ifes.edu.br*

Abstract

Gas consumption is a critical aspect of the pelletizing process, directly influencing operational costs and environmental impact. This study investigates the application of a multivariate time series forecasting pipeline for predicting gas consumption in pelletizing plants. The pipeline comprises: (i) data preprocessing, (ii) converting the dataset into a tabular format using a sliding window technique, (iii) applying feature selection methods, and (iv) employing machine learning tuned via AutoML. The methodology was tested on a dataset with 45 operational parameters collected over 90 days from an industrial plant, with predictions evaluated using Root Mean Squared Error (RMSE). In step (iii), twelve features were identified as the most relevant based on the RF importance index. In the final stage, two AutoML approaches were employed: neural architecture search using AutoKeras and the DEAP (Distributed Evolutionary Algorithm) framework. The neural network architectures tested included MLP, RNN, LSTM, and Conv1D. The best performance was achieved by the DEAP framework combined with LSTM networks, which yielded an RMSE of 0.33. Although AutoML did not outperform the statistical model in terms of RMSE values, regarding training time, AutoML models were significantly more efficient than the statistical approach, optimizing computational resource usage and enabling faster model adjustments. These findings confirm the generalization capability of the pipeline, demonstrating its applicability across different industrial environments.

Keywords: Neural Networks, Feature Selection, Machine Learning, AutoML, Evolutionary Algorithm.

1 Introduction

A univariate time series consists of a sequence of data points for a single variable, collected at regular intervals over time. This type of modeling enables the analysis of seasonality, growth rates, and trends, as well as the forecasting of a variable's future values based on its historical data [1]. A multivariate time series incorporates multiple dependent variables. In this model, each variable is influenced not only by its past values but also by the past values of other variables involved. This arrangement becomes particularly challenging when complex interdependencies and nonlinearities exist between the variables.

A commonly adopted pipeline for multivariate time series prediction involves four main stages, as outlined in the methodologies used by [2] and [3]: (i) data preprocessing, which includes cleaning the initial dataset, data normalization, removal of variables with missing information for extended periods, and elimination of constant variables; (ii) converting the processed dataset into a tabular format using a sliding window technique; (iii) feature selection, as described by [4]; and (iv) applying regression algorithms for modeling and forecasting, which can be tuned using AutoML. A machine learning (ML) pipeline is a structured workflow where a sequence of steps is applied to data to transform, train, and generate predictions or insights. Each step is interconnected, with the output of one step serving as the input for the next [5].

Previous research by [6] used the following the described pipeline for forecasting gas consumption in pelletizing plants, demonstrating its potential using data from a specific industrial furnace. However, a

key limitation of this study was the lack of validation across different plants, raising questions about the generalizability of the pipeline. This study addresses this gap by applying the pipeline to data from a furnace operated by another company, collected over 90 days, to evaluate its performance in a new context.

The production of iron ore pellets involves three main stages: raw material preparation, green pellet formation, and thermal processing. The thermal processing stage, carried out in moving grate furnaces, significantly impacts operational costs and energy efficiency, with natural gas consumption being a key factor. Natural gas accounts for approximately 10% of total production costs and represents 34% of the energy matrix costs in the pelletizing process [7]. Beyond the financial aspect, there are also environmental concerns associated with natural gas combustion. As highlighted by [8], approximately 20% of the carbon dioxide (CO_2) emissions across the entire production chain result from burning this fuel. Consequently, predicting gas consumption is a crucial step toward implementing measures to optimize its use.

The initial dataset undergoes a data cleaning process. Subsequently, feature selection is performed to identify the process variables most relevant to specific gas consumption, utilizing importance indices calculated by the algorithms. The Random Forest (RF) method was employed for feature selection [9]. The subset obtained through RF serves as input for the statistical VARMAX model (Vector Autoregressive Moving Average model with eXogenous variables) [10]. This result acts as the baseline, serving both as a proof of concept and as a standard for comparison with results generated later using machine learning methods. The analysis is conducted using the Root Mean Squared Error (RMSE) metric and training time.

Neural network models have shown promise in handling nonlinear relationships among variables in multivariate time series, which are not easily addressed by statistical models [11]. Achieving good results with neural networks depends on selecting the architecture most suitable for the problem, and numerous studies have focused on comparing and proposing different architectures for various tasks [12]. Although it is possible to optimize parameters through experimentation, studies like [13] argue that a trial-and-error approach is too costly for neural networks, which can involve hundreds of thousands of hyperparameters. AutoML (Automated Machine Learning) has been presented as one solution to this challenge. AutoML refers to a set of techniques and tools that automate the process of building machine learning models. AutoML tasks are primarily divided into two categories: NAS (Neural Architecture Search), which focuses on finding the best architecture; and hyperparameter optimization, which aims to identify the best hyperparameters for a given neural network architecture.

Two forms of AutoML were employed in this study: NAS using AutoKeras [14] and the DEAP framework (Distributed Evolutionary Algorithm) [15]. The neural networks chosen for evaluation included MLP (Multilayer Perceptron), RNN (Recurrent Neural Network), LSTM (Long Short-Term Memory), and Conv1D (1D Convolutional Neural Network). As a result, eight models were generated and evaluated. Finally, using statistical methods as a baseline reference, the various results obtained from machine learning methods leveraging hyperparameter and architecture optimization techniques will be compared. By analyzing the performance of AutoML techniques against statistical methods, it will be possible to evaluate whether the generalizability of the pipeline. Figure 1 illustrates the pipeline used in this study.

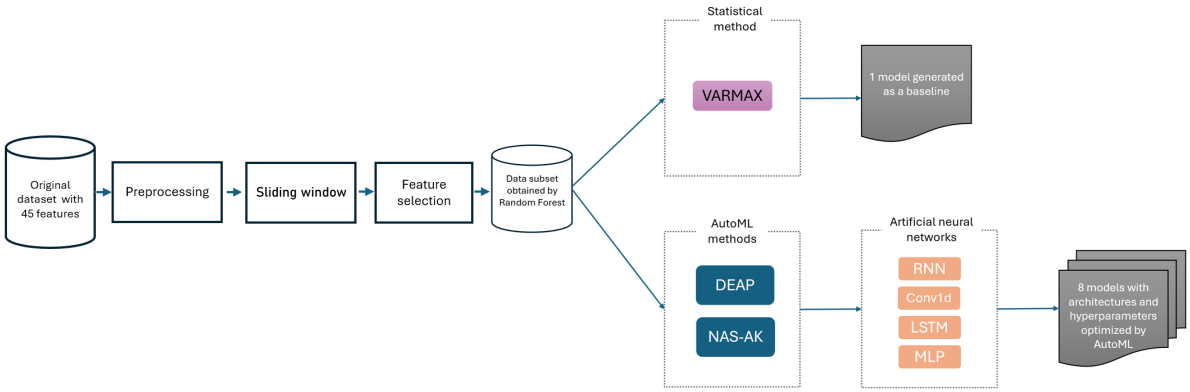

Figure 1: Pipeline for multivariate time series prediction using statistical method and neural networks with AutoML.

The following text are organized as follows: Section 2 presents related works; Section 3 describes the methods, metrics, and dataset investigated in this study; Section 4 presents the results and discussion; and the conclusions are provided in Section 5.

2 Related work

The methodology proposed by Gonzalez-Vidal, Jimenez, and Gomez-Skarmeta [2] focuses on optimizing multivariate time series forecasting through a systematic machine learning pipeline. The process begins with rigorous preprocessing, including handling missing data, scaling, and smoothing techniques to reduce noise. Feature selection is a critical step, where methods like Recursive Feature Elimination (RFE) and feature importance rankings from RF are employed to identify the most relevant variables. The pipeline uses a sliding window approach to convert temporal data into tabular form, enabling it to leverage algorithms such as Gradient Boosting, RF, and LSTM networks. A key strength of their approach lies in the inclusion of hyperparameter optimization using grid search and Bayesian optimization to maximize model performance. The methodology was validated using datasets from energy consumption forecasting in industrial plants. Specifically, the pipeline achieved a 13% reduction in RMSE compared to traditional ARIMA models and outperformed standard machine learning techniques like SVR and Decision Trees by 7%–10% in MAE and RMSE metrics, demonstrating robustness across varying conditions and datasets.

The methodology of Papadopoulos et al. [16] addresses the challenges of high-dimensional and noisy data in multivariate time series forecasting by integrating advanced preprocessing, feature selection, and hybrid modeling techniques. The preprocessing phase employs missing data imputation using K-Nearest Neighbors (KNN) and seasonal decomposition to capture both short- and long-term patterns. Dimensionality reduction techniques are applied to minimize redundancy and computational complexity. Feature selection combines mutual information and feature importance scores derived from Gradient Boosting algorithms. The modeling phase leverages hybrid architectures, combining machine learning models (e.g., SVR, RF) with deep learning techniques such as Temporal Convolutional Networks (TCN) and LSTM networks. Hyperparameter optimization is conducted using Genetic Algorithms, fine-tuning both model architecture and parameters for optimal performance. When tested on financial time series datasets, the proposed approach demonstrated a 1% reduction in RMSE compared to VARMAX and ARIMA. The methodology proved effective across different domains, highlighting its scalability and ability to generalize to diverse time series forecasting tasks.

Oliveira, Komati, and Andrade [6] proposed implementing models to forecast gas consumption in the pelletizing process, emphasizing financial optimization and reducing the environmental impact associated with gas combustion. The study was conducted at a plant in Serra, Espírito Santo, Brazil, using real-world data from 36 operational variables collected over 90 days at 4-hour intervals. The primary goal was to evaluate and compare neural networks optimized through AutoML techniques while investigating whether feature selection, following the application of a sliding window technique, would constitute a necessary step in the pipeline. The raw dataset underwent a cleaning process to remove inconsistencies before being subjected to classical statistical methods such as ARIMA, VARMAX, and GARCH. These methods established baseline performance metrics. The data were then transformed into a tabular format using the sliding window technique and subsequently passed through three feature selection methods: Pearson Correlation, Adaboost, and RF. The resulting datasets, including the one without feature selection, were used as inputs for three machine learning models (MLP, Adaboost, and RF), generating 12 combinations of datasets and models. Performance was evaluated using RMSE and execution time. The best-performing configuration at this stage served as the basis for the AutoML experiments. Four AutoML approaches were employed: NAS, including a custom implementation developed by the author, NAS with AutoKeras, DEAP, and NeuroEvolution of Augmenting Topologies (NEAT). These techniques were applied to explore and optimize hyperparameters and neural network architectures (MLP, RNN, LSTM, and 1D-CNN), resulting in a total of 11 tested configurations. The results demonstrated that AutoML-optimized neural networks outperformed the statistical methods, with the RNN model optimized by NEAT achieving a significant RMSE reduction from 0.7 (best statistical model) to 0.32. RF excelled among the feature selection methods, and MLP emerged as the best regressor when feature selection was applied. Although the results are promising, the study has limitations that pave the way for future research. First, the generalization of the models to different pelletizing plants or other industrial applications has not been extensively validated. Future studies could explore the applicability of these models in various industrial contexts, considering differences in data types and operational conditions.

The study of Melo, Andrade, and Komati [17] investigates the impact of the sequence between feature selection and sliding window techniques on multivariate time series forecasting, specifically for gas consumption prediction in iron ore pelletizing plants. The dataset consists of 45 operational variables collected hourly over 90 days. Two approaches were evaluated: (1) applying the sliding window technique before feature selection, and (2) applying feature selection prior to the sliding window transformation. Three feature selection methods were tested—Pearson Correlation, AdaBoost, and RF—combined with regression models including AdaBoost, RF, and MLP. Results showed that the sliding window-first approach effectively reduces dataset size while maintaining accuracy. Additionally, the study highlights the advantages of MLP in handling increased variable counts in the second approach, making it a robust choice for complex industrial datasets.

RF was chosen for feature selection due to its effectiveness in ranking feature importance and its robustness against noise and irrelevant variables. Previous studies [2, 6, 17] have demonstrated the advantages of RF-based feature selection in optimizing multivariate time series forecasting. Additionally, research has shown that RF enhances both model interpretability and computational efficiency [16].

VARMAX was chosen as the baseline model in this study due to its extensive use in time series forecasting. Papadopoulos et al. [16] demonstrated that statistical models like VARMAX serve as strong benchmarks for evaluating deep learning models in industrial applications. MLP, RNN, LSTM, and Conv1D were selected to enable direct comparison with Oliveira, Komati, and Andrade [6], who evaluated these models for multivariate time series forecasting of gas consumption. Using the same architectures ensures methodological consistency and facilitates performance assessment across different optimization strategies.

NAS with AutoKeras was employed due to its automated optimization capabilities. Truong et al. [18] compared several NAS frameworks, including AutoKeras, H2O-AutoML, TPOT, Auto-sklearn, Ludwig, and Darwin. Their findings indicate that AutoKeras offers stability across various tasks, with consistent performance, particularly in multiclass classification and regression problems. While its binary classification performance was inferior to some other frameworks, AutoKeras demonstrated rapid convergence and efficiency within moderate time constraints. Although its effectiveness varies depending on the dataset, it remains a competitive option for specific machine learning applications.

DEAP was selected for NAS due to its flexibility, efficiency in handling large and complex datasets, and its ability to incorporate evolutionary optimization techniques [19]. Unlike traditional NAS approaches that rely on grid search or Bayesian optimization, DEAP leverages Genetic Programming (GP) and Evolutionary Algorithms (EAs) to explore a diverse range of neural network architectures and hyperparameters.

3 Materials and methods

The evaluation metric chosen for this study was RMSE, which is a commonly used measure to assess the accuracy of predictive models, particularly in regression tasks. This metric quantifies the discrepancy between the values predicted by the model and the actual values, reflecting the average magnitude of the error. The closer the RMSE value is to zero, the better the prediction performance. The RMSE is calculated by taking the average difference between predicted and actual values, squaring these differences, and then applying the square root. Mathematically, RMSE is expressed as shown in Equation 1.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i' - y_i)^2} \tag{1}$$

where y_i' represents the predicted values from the model, y_i the observed values, and N the total number of observations. This metric is particularly sensitive to outliers, as larger errors are significantly penalized due to the squaring of the differences.

3.1 Data and preprocessing

This study analyzed operational data collected over 90 consecutive days, from April 15 to July 15, 2023, at a pelletizing plant in Espírito Santo, Brazil. The data were recorded hourly, resulting in a dataset of 2,208 raw records and 45 operational variables. The variables were selected based on expert knowledge and their relevance to the pelletizing process and include:

- Granulometry percentages of the pellet bed (10 variables): Representing the size distribution of pellets, these variables affect airflow and heat transfer efficiency during combustion.
- Fan speeds (12 variables): Measured in rotations per minute (RPM), these control the air circulation across different process stages, including drying, firing, and cooling. Each fan is equipped with a valve known as a Damper, which regulates the airflow.
- Pressure measurements (5 variables): Captured at critical process points to monitor and regulate combustion dynamics.
- Temperature metrics (3 variables): Including drying temperature, firing temperature, and cooling temperature, these directly influence the efficiency of thermal processes.
- Thermal profile and mean firing temperature (2 variables): Key indicators of the thermal stability and uniformity of the pelletizing process.
- Pellet bed height (1 variable): Influences airflow resistance and heat distribution during firing.

- Grate car speed (1 variable): Affects the exposure time of pellets to the heat source.
- Pellet type and fixed carbon content (2 variables): Reflect the quality and combustion properties of the pellets being processed.
- Additives and feed composition (5 variables): Includes quantities of limestone, material input, and return material, as well as pellet feed characteristics, which impact chemical composition and process stability.
- Production metrics (2 variables): Hourly production rate and total return of pellets, representing operational efficiency.
- Loss factor (1 variable): Used in the transformation from raw ore to sintered ore.
- Gas consumption (1 variable): The target variable, measured in cubic meters per hour (m^3/h), which reflects the energy input for the process.

Records containing granulometry variables with negative values or whose sum exceeded 100% were discarded. Additionally, inconsistencies in pellet production data were identified: values below 900 or above 1,400 tons per hour (t/h) were deemed outliers by process experts, as they do not reflect actual production rates. These records were also excluded, leaving a preprocessed dataset with 2,178 valid records.

3.2 Sliding window technique

Seasonality in time series refers to cyclical and predictable patterns that recur at specific intervals, such as days, weeks, months, or years. A time series $y(t)$ can be decomposed into three components, $y(t) = T(t) + S(t) + R(t)$, where $T(t)$ represents the *trend* component, $S(t)$ corresponds to the *seasonal* component, and $R(t)$ denotes the *residual* component, typically characterized as a stochastic signal in time series analysis.

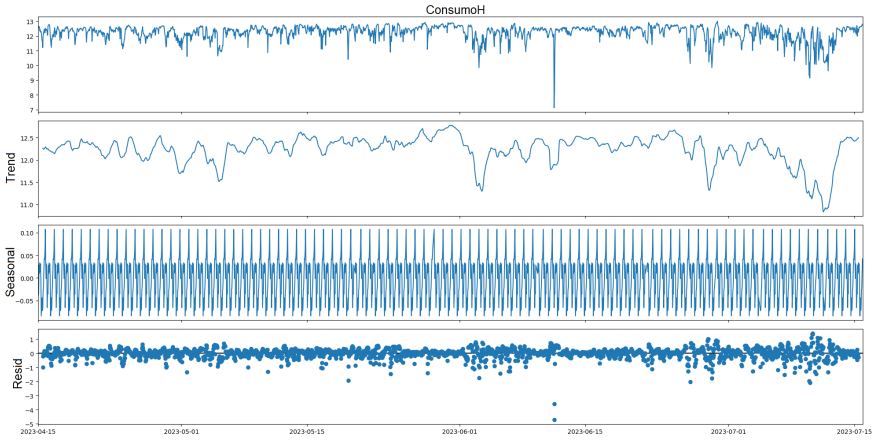

Figure 2: Seasonal decomposition of gas consumption (target variable).

Figure 2 presents the seasonal decomposition of the time series for actual gas consumption. The uppermost plot shows variations in gas consumption itself, represented by the target variable "ConsumoH". The "Trend" plot captures long-term movements, while the "Seasonal" plot reflects periodic fluctuations, and the "Resid" plot represents the residual component. The seasonal component exhibits low values, generally within the decimal range, indicating minimal influence on overall variability. In contrast, the trend component shows more pronounced oscillations, suggesting a significant role in data variation and possible non-stationarity in the time series. To verify stationarity, the Augmented Dickey-Fuller Test was applied, which evaluates the degree of stationarity in the series. The test revealed a high degree of stationarity, as indicated by an extremely low 9.053×10^{-13}, confirming that the series is stationary.

Time series data is converted into a tabular format using a sliding window of size N, where each window of N consecutive observations forms a row in the dataset [20]. The hyperparameter N determines how the data is structured. For seasonality analysis, N was set to 24 periods (approximately 24 hours), representing a full daily production cycle, including typical process adjustments. Applying this window to each of the 45 variables results in a transformed dataset with 1,080 variables, obtained by multiplying the 45 original variables by the 24-period window.

3.3 Feature selection

Establishing criteria for forming feature subsets from a preprocessed dataset aims primarily to prevent overfitting, reduce processing time, and enhance model performance in target variable prediction [21]. The RF method, an ensemble of decision trees, was used to assess the relative importance of each feature in relation to the target variable. This approach provides a quantitative basis for identifying and prioritizing the most relevant attributes for the predictive model. The importance of a variable is determined by averaging the contributions assigned by all the individual trees within the forest [22].

The RF algorithm was configured with the maximum number of features set to 98%, while the remaining parameters were kept at their default values in the Scikit-Learn library. These default values include: 100 estimators, the mean squared error criterion for split selection, a minimum of 2 samples required to split a node, at least 1 sample per leaf, no restrictions on the minimum weight fraction of leaf nodes, unlimited depth for leaf nodes, no minimum impurity decrease applied, bootstrap sampling enabled, no scaling applied to random sampling, and no parallelization or reuse of previous solutions.

After creating the feature selection subset, the data were subjected to a normalization process. Normalization ensures that variables have comparable scales, significantly contributing to improved model performance. In this study, the standardization technique was applied, $z = \frac{x-\mu}{s}$. This formula adjusts each sample (x) by subtracting the mean (μ) and dividing by the standard deviation (s). The result is a transformation where the data distribution has a mean of 0 and a standard deviation of 1, eliminating the effects of differing scales among variables.

In the feature selection step, the result was a subset of 12 features, obtained based on the importance index provided by the RF algorithm.

- The gas consumption from the three previous periods, representing the amount of gas used in pellet burning one, two, and three hours prior to the current evaluation;
- Temperatures:
  - The actual thermal profile temperatures of the furnace corresponding to process zones (drying, heating, and heat recovery);
  - The thermal profile temperature of the previous hour, based on the current process record;
  - The actual recovery duct temperature, which supplies hot gases obtained during the cooling phase to be mixed with other gases in earlier furnace stages;
  - The hourly average pellet burning temperature;
- The actual revolutions per minute (RPM) of fans 1 and 2, which control air circulation in the traveling grate furnace;
- The actual speed of the grate (m/min) where the pellets move through the furnace;
- The actual pressure in the downdraft drying hood, which is the second and final phase of pellet drying in the furnace;
- The hourly production of fired pellets exiting the pelletizing furnaces.

Identifying the variables that most influence gas consumption over time enables process controllers to make more precise adjustments to the involved parameters. This approach allows for more efficient gas consumption reduction by adopting targeted actions that minimize the risk of ineffective or even detrimental interventions, both for the process and the environment.

3.4 Machine learning techniques

Based on the subset defined by feature selection, two approaches were implemented for model construction and evaluation: ML methods and a statistical method. The statistical method chosen to create the baseline model for comparison with other techniques was the multivariate VARMAX model. The other investigative path involved applying AutoML methods to optimize models and their results. Two primary AutoML techniques were employed: DEAP and NAS via AutoKeras (referred to as NAS-AK). These methods were applied to different regression algorithms for gas consumption prediction: RNN, Conv1d, LSTM, and MLP.

AutoML is described as the task of finding the function f that best generalizes to any dataset T with minimal human intervention [23]. The function f can be a composition of multiple functions, including input space transformations, data sampling, predictor combinations, and other processes. The function

$$f(x) = \nu_{\theta_\nu}(\phi_{\theta_\phi}(x)) \tag{2}$$

represents a complete model or pipeline, where ν is a classification model and ϕ is a feature transformation methodology, with θ_ν and θ_ϕ being the corresponding hyperparameters. Each of these components may itself consist of other functions or models.

In this context, AutoML is understood as the search for the functions ν and ϕ, along with their hyperparameters θ_ν and θ_ϕ, using the dataset T. This study applies AutoML specifically to the classification model ν and its hyperparameters θ_ν, Considering that ϕ_{θ_ϕ} has already been evaluated during the feature selection process.

3.4.1 Auto-Keras

AutoKeras[1] was developed by the DATA Lab (Texas A&M University) and is based on the Keras module, referred to by its developers as AK. The AK library allows users to set up neural networks with minimal lines of code using the Efficient Neural Architecture Search (ENAS) approach, which employs an algorithm to search for a subgraph within a set of graphs. The AK library offers a variety of models, including those used in this study: MLP, LSTM, RNN, and Conv1d.

Table 1 is organized into two sections. The upper section presents parameters related to the neural network structure, such as the number of hidden layers, neurons, blocks, and filters. These parameters were defined by the author, with empirically determined ranges, to be explored by AutoKeras. The lower section lists hyperparameters whose tuning was delegated to AutoKeras. These include values commonly considered by the tool, such as different activation functions, optimizers, and dropout rates, which are automatically adjusted to improve the model's performance, as described in the AutoKeras documentation[1].

Table 1: Parameters and values used in the NAS-AK method.

Parameters	values
Number of Hidden Layers	[0,.., 512]
Number of Neurons	[0,.., 128]
Blocks	[0,.., 5]
Filters	[32,.., 128]
Activation Function	[ReLU, Tanh, Sigmoid, Softmax, ELU, LReLU]
Learning Rate	[0,1, 0.01, 0.001]
Optimizer	[Adam, RMSprop, SGD]
Dropout Rate	[0.0,.., 0.5]

3.4.2 DEAP

DEAP is a framework for developing custom Evolutionary Algorithms (EAs), written in Python [15]. Neuroevolution is a term encompassing strategies to optimize weights, architectures, and hyperparameters of neural networks through evolutionary computation, aiming to maximize a chosen performance metric, also known as the fitness function [24]. In this study, the data structure chosen for DEAP was a list containing the parameters used to configure the models. Each individual's genes were represented by the elements of this list, forming the individual to be evaluated. The fitness function was defined as the minimization of RMSE. The genome consisted of parameters, as shown in Table 2.

Table 2: Parameters and genome values used in the DEAP framework.

Parameters	Genome Values
Number of Hidden Layers	[0,.., 512]
Learning Rate	[0.01,.., 1×10^{-8}]
Activation Function	[ReLU, ELU, Tanh, Sigmoid, SELU, Softmax, GELU, Mish, PReLU]
Learning Rate	[0.01,.., 1×10^{-8}]
Optimizer	[Adam, RMSprop, ASGD, NAdam, RAdam, Adamax, SGD, Adadelta, Adagrad]
Dropout Rate	[0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5]

[1] `https://autokeras.com/`

The standard implementation of the DEAP library, available in its official GitHub repository[2], does not allow developers to directly define either the number of neurons in the layers or the number of blocks in the architecture. Instead, this configuration is handled internally by the library.

The parameters for the evolutionary process were defined as follows: the pairing criterion was a two-point crossover, where the genes of individuals are mixed to form two new individuals, ensuring that both have the same number of genes. The chosen mutation method was "Shuffle Indexes", where, based on a random criterion, one gene in the individual's parameter list is selected and swapped with the original gene from the parent individuals [25]. The probability of this mutation occurring was set to 5%. At the end of the process, the individual with the best performance in minimizing the fitness function for each model was selected. An example of genome crossover and mutation is illustrated below. Consider two parent genomes:

- Parent Genome 1: [64, 0.1, 'tanh', 0.001, 'Adam', 0.25]
- Parent Genome 2: [128, 0.001, 'sigmoid', 0.01, 'SGD', 0.20]

Applying a two-point crossover results in two offspring, where two genes are swapped between the parents. The exchanged genes are indicated in bold:

- Offspring Genome 1: [64, 0.1, **'sigmoid'**, 0.001, **'SGD'**, 0.25]
- Offspring Genome 2: [128, 0.001, **'tanh'**, 0.01, **'Adam'**, 0.20]

Suppose a mutation occurs in Offspring Genome 2. One of the genes is altered, as shown in the example below, with the mutated gene highlighted in bold:

- Mutated Offspring Genome: [**32**, 0.001, 'tanh', 0.01, 'Adam', 0.20]

4 Experiments and results

The experiments were conducted using Google Colab, a hosted Jupyter Notebook service that requires no setup and offers access to computing resources, including GPUs and TPUs.

4.1 Statistical Method Result

The VARMAX model, configured with $p = 3$ (three autoregressive lags) and $q = 1$ (one moving average term), achieved the following results: RMSE = 0.25 and time taken by the VARMAX model to generate the output = $12 : 31 : 53$, with the results obtained through expanding window validation. This approach allows the model training to be performed progressively, with the training set growing over time as new data becomes available.

4.2 ML with AutoML Results

All models were trained and tuned using 75% of the dataset, while the remaining 25% was used for prediction testing. For training the regression models with DEAP, a population of 10 individuals was used across 40 generations. In the case of NAS-AK, the number of epochs was set to 200, and the maximum number of attempts to find the best model configuration during the training and optimization process was limited to 10 attempts.

The best parameters and configurations for each method and model are presented in Table 3 for the models applied using the DEAP method and in Table 4 for those applied using the NAS-AK method. The tables include specific model configurations, such as the number of layers, learning rate, activation function, optimizer, and dropout rate. Table 4 is divided into two parts: the first details the hyperparameters affecting the architecture structure, with values explicitly defined for optimization as shown in Table 1. The second part presents the parameters adjusted by Auto-Keras's internal optimization process.

Table 5 presents the method/model used in the first column, the RMSE achieved in the second column, and the training time in the third column, displayed in hours, minutes, and seconds (hh:mm:ss). The rows are ordered by method, with the first four rows corresponding to DEAP, followed by the results of NAS-AK. Bold values indicate the models with the lowest RMSE. The analyses were conducted in the Google Colab Pro environment, utilizing the A100 GPU as a hardware accelerator to enhance processing efficiency. The best RMSE result was achieved by the LSTM model using the DEAP method, with a value of **0.33** and a training time of **approximately 30 minutes**. This RMSE was the lowest among all combinations presented in the table, indicating that the model provided the most accurate predictions compared to the others.

[2] `https://github.com/deap/deap`

Table 3: Optimal parameters identified by the DEAP method for each neural network.

Hyperparameters	MLP	LSTM	Conv1d	RNN
Number of Layers	4	256	4	64
Learning Rate	1×10^{-6}	0,001	0,001	0,001
Activation Function	Selu	Sigmoid	Gelu	Selu
Learning Rate	1×10^{-7}	0,0001	1×10^{-5}	0,0001
Optimizer	Adadelta	RMSprop	Adagrad	RMSprop
Taxa de dropout	0,3	0,35	0,1	0,25

Table 4: Optimal parameters identified by the NAS-AK method for each neural network.

Hyperparameters	MLP	LSTM	Conv1d	RNN
Number of Layers	4	6	6	6
Number of Neurons	12	[12, 32]	[12, 32, 64]	[12, 32]
Blocks	-	-	-	2
Filters	-	-	-	[32, 64]
Learning Rate	0,001	0,001	0,001	0,1
Activation Function	Relu	Relu	Relu	Relu
Otimizador	Adam	Adam	Adam	Adam
Dropout Rate	0,25	0,0	0,0	0,0

Table 5: RMSE of test dataset and Training Time Results.

AutoML/Model	RMSE of test dataset	Training Time
DEAP/MLP	0,89	00:04:15
DEAP/LSTM	**0,33**	**00:29:31**
DEAP/Conv1d	**0,35**	**00:33:54**
DEAP/RNN	0,41	00:07:24
NAS-AK/MLP	**0,35**	**00:02:03**
NAS-AK/LSTM	0,48	00:11:44
NAS-AK/Conv1d	**0,38**	**00:02:05**
NAS-AK/RNN	0,55	00:05:19
VARMAX	**0,25**	**12:31:53**

Although VARMAX achieved the lowest RMSE, it was also the method with the longest response time, exceeding 12 hours. The machine learning-based model, in comparison, achieved a mean RMSE of 0.35 with only 2 minutes of training. Overall, the training time for the best results obtained using the DEAP library (approximately 30 minutes) was considerably longer than that of the best-performing models found by NAS-AK (around 2 minutes). This suggests a trade-off between accuracy and training time, where the choice of model and methodology should take into account the specific requirements of the problem at hand.

Another way to qualitatively visualize the results is through a scatter plot and a residuals histogram. The scatter plot displays points based on the predicted and actual values. The less dispersed the points, the better the model's performance; in other words, a more concentrated dispersion indicates that the model is making more accurate predictions with smaller errors. The residuals histogram complements this analysis by providing insights into the distribution of prediction errors. A well-behaved histogram, with a distribution close to a normal curve centered around zero, suggests that the model has smaller errors. Conversely, a histogram with asymmetries or peaks away from the center indicates larger prediction errors.

Figure 3 illustrates the difference between the results of the LSTM (left) and MLP (right) algorithms, which achieved RMSE values of 0.33 and 0.89, respectively. The LSTM model demonstrated the best performance, while the MLP model produced the worst result among the AutoML technique models. In the left graphic, the scatter plot reveals a pronounced concentration of points in the upper right quadrant, indicating more accurate predictions. This concentration is more evident compared to the dispersion of points along the diagonal line. Additionally, the residuals histogram shows significant centralization near zero, suggesting smaller errors between the actual and predicted values. Conversely, in the right, there is a greater dispersion of points, highlighting the inferior performance of the MLP model. The residuals histogram displays peaks away from zero, reflecting a higher margin of error in predictions. The presence of peaks in negative values indicates that the predicted values are generally lower than the actual ones.

We also present the forecasting charts in Figure 4, using a direct multiple-step forecast approach. A

forecasting chart displays past data over a specific period and includes a trend line that extends beyond the historical data. This trend line represents the prediction and includes upper and lower bounds, which form the confidence interval. In this case, the confidence interval was set to 2 standard deviations, a commonly used threshold for identifying outliers [26].

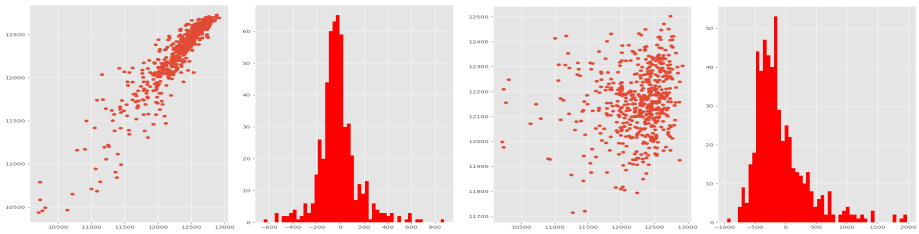

Figure 3: Scatter plot and residuals histogram for the DEAP/LSTM model (left) and DEAP/MLP model (right).

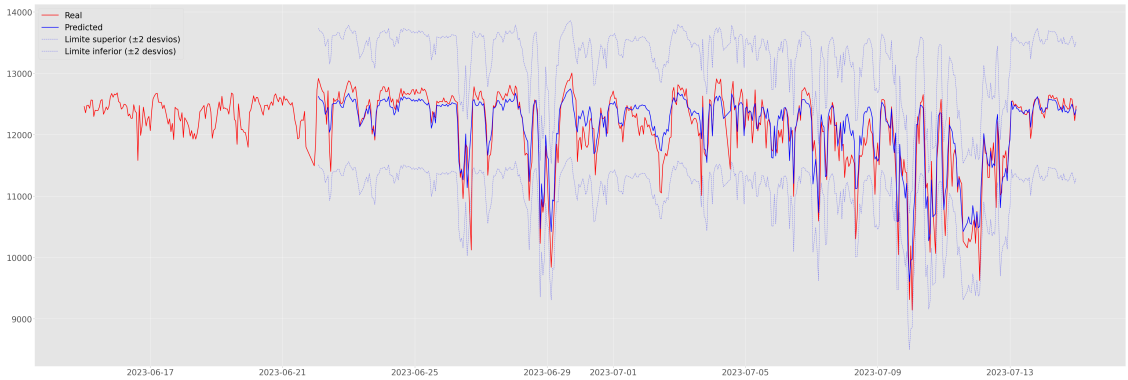

Figure 4: Projection plot generated by the DEAP/LSTM model starting from 06/15/2023.

In this study, the best AutoML result was achieved with DEAP/LSTM, obtaining an RMSE of 0.33, whereas in Oliveira, Komati, and Andrade [6] work, the best RMSE of 0.32 was obtained with NEAT/RNN. Both studies similarly identified that MLP performs well and effectively captures complex patterns. Regardless of the AutoML-based method used, MLP demonstrated a favorable trade-off between training/execution time and computational cost compared to other models.

5 Conclusion

This study investigated the generalization and performance of machine learning pipelines for gas consumption forecasting in industrial processes, using multivariate time series data. The analysis focused on evaluating the generalization capability of the pipeline developed by [6] when applied to data from a different pelletizing plant. Specifically, data from a Samarco Mineração plant located in Espírito Santo were used, while the original dataset was obtained from other plant. Although the plants differ, they share structural similarities in their pelletizing furnaces.

The results presented in this study are relevant from both an industrial and an experimental perspective. From an industrial standpoint, predicting gas consumption in the pelletization process is crucial, as natural gas accounts for approximately 10% of total production costs and 34% of the energy matrix costs in pelletizing plants. Furthermore, optimizing gas consumption directly contributes to cost reduction and environmental sustainability by minimizing CO_2 emissions. The predictive models developed in this study provide insights that can support decision-making in industrial operations, offering a means to enhance energy efficiency and reduce operational expenses.

From an experimental perspective, this study investigates the generalization capability of a machine learning pipeline for gas consumption forecasting across different industrial plants. The findings validate the applicability of AutoML techniques and feature selection methodologies in real-world industrial settings. Although the AutoML-based models did not outperform the statistical baseline in terms of RMSE, they demonstrated significant advantages in computational efficiency, allowing for faster training and adaptation to new operational conditions. This aspect is particularly relevant for industrial applications where rapid adjustments to changing conditions are necessary.

For future studies, the implementation of the investigated pipeline in a real industrial environment is planned, allowing for the evaluation of its generalization and performance in a continuous and dynamic context. This implementation will not only validate the results obtained in controlled scenarios but also uncover additional challenges and opportunities associated with deploying the pipeline under real operational conditions.

An additional avenue of exploration could be investigating the impact of incorporating external variables, such as climatic or market indicators, on the pipeline's performance. This approach could expand its applicability across diverse industrial scenarios. Another promising direction is the use of continuous learning techniques, which enable models to adapt to process changes over time without requiring costly and exhaustive retraining. This is particularly relevant in dynamic industrial processes, where patterns may shift due to operational or seasonal changes. Finally, integrating predictive systems with industrial automation tools, such as Digital Twins, could pave the way for more comprehensive solutions, linking gas consumption forecasting directly to automated control and process optimization strategies. Such integration could amplify financial and environmental gains by aligning predictions with automated corrective actions.

Acknowledgment

Professor Komati thanks CNPq for DT-2 grant (302726/2023-3) and project 407742/2022-0, also thanks Fundação de Amparo à Pesquisa e Inovação do Espírito Santo for project 1023/2022 P:2022-8TZV6.

References

[1] M. Mudelsee, "Trend analysis of climate time series: A review of methods," *Earth-science reviews*, vol. 190, pp. 310–322, 2019. [Online]. Available: https://doi.org/10.1016/j.earscirev.2018.12.005

[2] A. Gonzalez-Vidal, F. Jimenez, and A. F. Gomez-Skarmeta, "A methodology for energy multivariate time series forecasting in smart buildings based on feature selection," *Energy and Buildings*, vol. 196, pp. 71–82, 2019. [Online]. Available: https://doi.org/10.1016/j.enbuild.2019.05.021

[3] X. Yuan, J. Yuan, T. Jiang, and Q. U. Ain, "Integrated long-term stock selection models based on feature selection and machine learning algorithms for China stock market," *IEEE Access*, vol. 8, pp. 22 672–22 685, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.2969293

[4] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," *Journal of Applied Science and Technology Trends*, vol. 1, no. 1, pp. 56–70, 2020. [Online]. Available: https://doi.org/10.38094/jastt1224

[5] S. Meisenbacher, M. Turowski, K. Phipps, M. Rätz, D. Müller, V. Hagenmeyer, and R. Mikut, "Review of automated time series forecasting pipelines," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 12, no. 6, p. e1475, 2022. [Online]. Available: https://doi.org/10.1002/widm.1475

[6] V. M. de Oliveira, K. S. Komati, and J. O. Andrade, "Implementing neuroevolution for gas consumption forecasting in the steel industry," in *2024 L Latin American Computer Conference (CLEI)*, 2024, pp. 1–10. [Online]. Available: https://doi.org/10.1109/CLEI64178.2024.10700389

[7] A. F. F. d. Cruz, E. B. Haagensen, L. R. Zanon, W. W. D. Regattieri, and M. M. Machado, "Aumento da eficiência energética em fornos de pelotização," in *Anais do 6° Simpósio Brasileiro de Aglomeração de Minérios*, 2018, pp. 127–139. [Online]. Available: https://doi.org/10.5151/2594-357X-31608

[8] E. M. d. Santos, M. T. W. Fagá, C. B. Barufi, and P. L. Poulallion, "Gás natural: a construção de uma nova civilização," *Estudos Avançados*, vol. 21, pp. 67–90, 2007. [Online]. Available: https://doi.org/10.1590/S0103-40142007000100007

[9] V. Pavithra and V. Jayalakshmi, "Hybrid feature selection technique for prediction of cardiovascular diseases," *Materials Today: Proceedings*, vol. 81, pp. 336–340, 2023, international Virtual Conference on Sustainable Materials (IVCSM-2k20). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214785321022549

[10] A. Garcia-Hiernaux, J. Casals, and M. Jerez, "Fast estimation methods for time-series models in state–space form," *Journal of Statistical Computation and Simulation*, vol. 79, no. 2, pp. 121–134, 2009. [Online]. Available: https://doi.org/10.1080/00949650701617249

[11] R. Wan, S. Mei, J. Wang, M. Liu, and F. Yang, "Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting," *Electronics*, vol. 8, no. 8, p. 876, 2019. [Online]. Available: https://doi.org/10.3390/electronics8080876

[12] D. Baymurzina, E. Golikov, and M. Burtsev, "A review of neural architecture search," *Neurocomputing*, vol. 474, pp. 82–93, 2022. [Online]. Available: https://doi.org/10.1016/j.neucom.2021.12.014

[13] R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan, N. Duffy *et al.*, "Evolving deep neural networks," in *Artificial Intelligence in the Age of Neural Networks and Brain Computing*. Elsevier, 2019, pp. 293–312. [Online]. Available: https://doi.org/10.1016/B978-0-323-96104-2.00002-6

[14] H. Jin, Q. Song, and X. Hu, "Auto-keras: An efficient neural architecture search system," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 1946–1956. [Online]. Available: https://doi.org/10.1145/3292500.3330648

[15] F.-M. De Rainville, F.-A. Fortin, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP – enabling nimbler evolutions," *ACM SIGEVOlution*, vol. 6, no. 2, pp. 17–26, 02 2014. [Online]. Available: https://doi.org/10.1145/2597453.2597455

[16] D. N. Papadopoulos, F. D. Javan, B. Najafi, A. H. Mamaghani, and F. Rinaldi, "Handling complete short-term data logging failure in smart buildings: Machine learning based forecasting pipelines with sliding-window training scheme," *Energy and Buildings*, vol. 301, p. 113694, 2023. [Online]. Available: https://doi.org/10.1016/j.enbuild.2023.113694

[17] T. P. Melo, J. O. Andrade, and K. S. Komati, "Impacto da ordem entre a seleção de características e a janela deslizante em séries temporais multivariadas na previsão do consumo de gás na pelotização de minério de ferro," in *Anais do XXV Congresso Brasileiro de Automática (CBA 2024)*. Rio de Janeiro, RJ: SBA, Sociedade Brasileira de Automação, 2024. [Online]. Available: https://www.sba.org.br/cba2024/papers/paper_9730.pdf

[18] A. Truong, A. Walters, J. Goodsitt, K. Hines, C. B. Bruss, and R. Farivar, "Towards automated machine learning: Evaluation and comparison of automl approaches and tools," in *2019 IEEE 31st international conference on tools with artificial intelligence (ICTAI)*. IEEE, 2019, pp. 1471–1479. [Online]. Available: https://doi.org/10.1109/ICTAI.2019.00209

[19] S. B. Hamida and G. Benjelloun, "Extending deap with active sampling for evolutionary supervised learning." in *ICSOFT*, 2021, pp. 574–582.

[20] B. Dhariyal, T. Le Nguyen, and G. Ifrim, "Back to basics: A sanity check on modern time series classification algorithms," in *International Workshop on Advanced Analytics and Learning on Temporal Data*. Springer, 2023, pp. 205–229. [Online]. Available: https://doi.org/10.1007/978-3-031-49896-1_14

[21] M. R. Islam, A. A. Lima, S. C. Das, M. F. Mridha, A. R. Prodeep, and Y. Watanobe, "A comprehensive survey on the process, methods, evaluation, and challenges of feature selection," *IEEE Access*, vol. 10, pp. 99 595–99 632, 2022. [Online]. Available: https://doi.org/10.1109/ACCESS.2022.3205618

[22] M. Haddouchi and A. Berrado, "A survey and taxonomy of methods interpreting random forest models," *arXiv preprint arXiv:2407.12759*, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2407.12759

[23] H. J. Escalante, "Automated machine learning—a brief review at the end of the early years," *Automated Design of Machine Learning and Search Algorithms*, pp. 11–28, 2021. [Online]. Available: https://doi.org/10.1007/978-3-030-72069-8_2

[24] A. Baldominos, Y. Saez, and P. Isasi, "On the automated, evolutionary design of neural networks: past, present, and future," *Neural Computing and Applications*, p. 519–545, 2020. [Online]. Available: https://doi.org/10.1007/s00521-019-04160-6

[25] C. Ludick and Q. Van Heerden, "A multi-objective optimization approach for disaggregating employment data," *Geographical Analysis*, vol. 55, no. 2, pp. 300–324, 2023.

[26] B. Qu, P. Liao, and Y. Huang, "Outlier detection and forecasting for bridge health monitoring based on time series intervention analysis." *Structural Durability & Health Monitoring (SDHM)*, vol. 16, no. 4, 2022. [Online]. Available: https://doi.org/10.32604/sdhm.2022.021446