Machine Learning-based Forecasting of Sensor Data for Enhanced Environmental Sensing

MARTA NARIGINA, ARTURS KEMPELIS, ANDREJS ROMANOVS
Department of Modeling and Simulation,
Riga Technical University,
Riga,
LATVIA

Abstract: - This article presents a study that explores forecasting methods for multivariate time series data, which was collected from sensors monitoring CO_2, temperature, and humidity. The article covers the preprocessing stages, such as dealing with missing values, data normalization, and organizing the time-series data into a suitable format for the model. This study aimed to evaluate Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), Vector Autoregressive (VAR) models, Artificial Neural Networks (ANNs), and Random Forest performance in terms of forecasting different environmental dataset parameters. After implementing and testing fifteen different sensor forecast model combinations, it was concluded that the Long Short-Term Memory and Vector Autoregression models produced the most accurate results. The highest accuracy for all models was achieved when forecasting temperature data with CO_2 and humidity as inputs. The least accurate models forecasted CO_2 levels based on temperature and humidity.

Key-Words: - Forecasting, Sensor Data, Machine Learning, Deep Learning, Neural Networks

Received: August 26, 2022. Revised: April 15, 2023. Accepted: May 3, 2023. Published: May 29, 2023.

1 Introduction

Various environments, such as medical facilities or agricultural settings, necessitate monitoring, typically accomplished using different sensor devices, [16]. However, deploying these sensor devices in each environment is often impractical. In situations where only a limited number of sensor devices are available, machine learning-based models can be employed to estimate other sensor values with a certain level of accuracy, thereby potentially serving as substitutes for physical devices.

This article explores a range of sensor forecasting techniques, explicitly addressing the assessment of multivariate time series forecast methods in environmental sensing. In this analysis, the forecasting capabilities were evaluated by comparing forecasted sensor values to actual sensor readings to measure the accuracy of the data forecasts. The method of developing forecasting models was used to forecast one of the three dataset parameters. This method was used not only to experimentally determine which of the chosen models had the highest accuracy but also to determine which parameter or sensor has the potential to be substituted with a forecast from sensor data trained machine learning model.

The dataset that was used for training the models contains 20560 entries that were gathered from CO_2 (ppm), Temperature (°C), and Humidity (%RH) sensors for one month, [1]. The results of the model experiments and data processing are discussed in this article.

Long Short-Term Memory (LSTM) networks, along with Convolutional Neural Networks (CNNs), Vector Autoregressive (VAR) model forecast, Artificial Neural Networks (ANN), and Random Forest, were used to test the forecast accuracy. To use the data, the following preprocessing steps of raw sensor data were included:

- Measures to address missing values.
- Ensuring temperature, humidity, and CO_2 readings were normalized.
- Presenting time-series data in the correct model format.

An initial model structure utilized CNN, incorporating a single convolutional layer and an assortment of pooled and fully established layers. Using 1D convolutions, the CNN model was effective in detecting localized patterns in the input data, while the pooling layers worked to decrease spatial proportions and computational complexity. Patterns were learned by utilizing post-convolutional and fully connected layers with ReLU activation, facilitating the introduction of nonlinearity. The sensor measurements' mean squared error was predicted and compared for the CNN and LSTM.

The initial steps to prepare the VAR model included handling missing values and assessing stationarity. Determining the optimal lag order, which dictates how many previous time steps should be factored into predictions, was accomplished via the Akaike Information Criterion (AIC). After monitoring residuals, any signs of non-stationarity were handled by implementing diverse transformations or differencing techniques.

2 Sensor Data Forecasting Methods

Nowadays, there exist many sensor data forecasting methods. Some standard forecasting methods are:

1. Linear regression;
2. Random Forest;
3. Artificial Neural Networks;
4. Support Vector Machines;
5. Hybrid approach.

2.1 Linear Regression

Linear regression is a widely adopted statistical method for predicting sensor data, relying on assuming a linear relationship between input and output variables, [2]. This method involves constructing a linear regression model by fitting a linear equation to the input data, which can be used to forecast the output variable, [3]. The model has the form:

$$y = \beta 0 + \beta 1x1 + \beta 2x2 + ... + \beta nxn + \varepsilon,$$

where y is the dependent variable, x1, x2, ..., xn are independent variables, β0, β1, β2, ..., βn are the model coefficients, and ε represents the error term. The objective of linear regression is to determine the optimal values of the coefficients that minimize the sum of the squared errors between the predicted and actual values in the training data, [2]. This process is accomplished through the least squares method, which seeks to obtain the values of the coefficients that minimize the sum of squared differences between the predicted and actual values, [2].

Once the model has been trained, it can be utilized to make predictions on new data by inserting the values of the independent variables into the equation and calculating the corresponding value of the dependent variable, [3]. However, when using linear regression for predicting sensor data, it is crucial to consider the presence of outliers or nonlinear relationships between the input and output variables. More complex models, such as random forests or neural networks, may be more appropriate, [4]. Additionally, the data should be preprocessed to handle missing values, scale the features appropriately, and handle any categorical variables using techniques such as one-hot encoding, [5].

2.1.1 Methodology for Implementing

Linear regression can be an efficient tool for analyzing data sets. The following methodology outlines how to implement it effectively.

Firstly, the dependent and independent variables must be identified. This will allow us to determine the correlation between them. Next, the most suitable regression model is chosen to align with the data. This will often depend on the relationship between the variables being analyzed. It is important to note that the data must be clean and outlier-free, [6].

Once the appropriate regression model has been identified, the regression equation coefficients can be estimated. This can be done through manual calculations or software such as Excel or R, [7]. After this, it is vital to check the accuracy of the regression equation. A useful metric for doing so is the R-squared value. This will give insight into how much of the variation in the dependent variable is explained by the independent variable, [2]. It is also useful to check for autocorrelation and heteroscedasticity, [5]. Finally, the model is tested using the data collected. This will show how well the model fits the dataset, [8].

To begin linear regression, one must first pinpoint the independent variables (alternatively referred to as predictors or features) that hold significance in predicting the dependent variable (alternatively referred to as the response variable or target variable) based on the gathered environmental sensor data, [2]. Following this, the next move is to instruct the linear regression model with a data set, optimizing its algorithm to triangulate the coefficients that will best diminish the contrast between the predicted and factual values. To conclude, for the near-final stage, a separate test set of data is applied to determine the model's level of precision and accuracy in correctly forecasting the dependent variable, [9].

2.1.2 Performance of the Algorithm in Terms of Accuracy and Precision

In terms of accuracy and precision, the algorithm's performance is noteworthy. To appraise the linear regression algorithm's effectiveness, one can employ a range of measurements, such as mean absolute error (MAE), mean squared error (MSE), and R-squared (R^2) coefficient, [2]. The MAE and MSE determine the mean disparity between calculated and actual values, whereas the R^2 coefficient determines the amount of independent variable variation that accounts for the dependent variable. The linear regression algorithm's precision and accuracy are hinged on the sufficiency and caliber of the environmental sensor data, coupled with the relevance of the independent variables adopted for the model, [7].

2.1.3 Limitations and Challenges

There exist some challenges and limitations to consider in this matter. In predicting environmental sensor data, linear regression is commonly utilized, yet various issues and constraints accompany it, [2]. Real-world environmental sensing applications may not always present a linear correlation between independent and dependent variables, which is an understandable limitation. Furthermore, it may not be competent in detecting intricate, nonlinear relationships within variables. Therefore, more advanced techniques, such as decision trees or neural networks, must be implemented, [3]. Lastly, the model's effectiveness is strongly influenced by the precision and soundness of the environmental sensor data used during the training process, [8].

2.2 Random Forest

Random Forest is a machine learning model that has gained popularity in predicting sensor data, [4]. An ensemble learning method combines multiple decision trees to make forecasts. Each decision tree is trained on a random subset of the input features and a random subset of the training data, and the final prediction is made by averaging the predictions of all the trees. Random Forest is particularly useful for handling complex interactions between input variables and can handle both continuous and categorical input variables and can be used for both regression and classification problems, [6].

To use Random Forest for sensor data forecasting, the data is first collected and preprocessed, which may include cleaning, normalization, and feature engineering, [3]. Then, the sensor data is divided into training and testing sets. The Random Forest model is trained on the training set using the input features and output variables, [5]. The model's performance is evaluated using the testing set, and metrics such as mean squared error (MSE) and R-squared are calculated, [4]. The model can be optimized by tuning the hyperparameters, such as the number of trees in the forest, the maximum depth of the trees, and the size of the random subsets, [3].

Random Forest has several advantages for predicting sensor data, including high accuracy, robustness, and the ability to provide information on the importance of each input feature, [10]. This information can help understand the underlying patterns in the data, [7]. It is important to note that Random Forest may not be suitable for all cases, and the presence of outliers or nonlinear relationships may require more complex models, such as neural networks or support vector machines, [11]. Additionally, data preprocessing techniques, such as handling missing values, scaling features appropriately, and one-hot encoding categorical variables, should be applied to improve the model's performance, [2].

2.2.1 Methodology for Implementing

Random Forest provides superior results by improving prediction precision and consistency using multiple decision trees in an ensemble learning algorithm, [4]. This environmental sensor data prediction tactic yields quantifiable gains in accurately forecasting air or water quality based on sensor outputs, [12].

To put the Random Forest model into practice, one must start by choosing the appropriate independent and dependent variables according to the data given by the environmental sensor, [4]. Next, the training phase of the model begins by taking a subset of the features and data points and using them to set up multiple decision trees, [4]. The impurity of these trees is evaluated to optimize the selection of thresholds and features used to split the data. The model is estimated using a different testing set to analyze and measure its precision and accuracy in predicting the dependent variable.

2.2.2 Performance of the Algorithm in Terms of Accuracy and Precision

Depending on the data being used, there are different ways to evaluate the Random Forest algorithm. Three standard metrics include the mean absolute error (MAE), mean squared error (MSE), and R-squared (R^2) coefficient, [4]. For this algorithm to be effective, it relies on accurate and plentiful data that can be measured by environmental sensors, [3]. Additionally, the choice of independent variables partially influences the model's success. If there are nonlinear correlations between the independent and dependent variables, Random Forest tends to be a better choice than linear regression, [4].

2.2.3 Limitations and Challenges

Several limitations and challenges exist for the Random Forest machine learning algorithm. One of these challenges is the potential for overfitting the training data resulting in poor generalization performance on the test data. It can also be computationally expensive when there is a significant number of features or data points, leading to limited scalability in some applications, [9]. Lastly, environmental sensor data quality and accuracy significantly affect the algorithm's performance, [6].

2.3 Artificial Neural Networks

Artificial Neural Networks (ANNs) are a popular machine learning technique for predicting sensor data inspired by the structure and function of the human brain. ANNs consist of interconnected nodes, or neurons, arranged in layers, which process information and make predictions. ANNs are particularly useful for handling complex and nonlinear relationships between input and output variables and can handle continuous and categorical input variables.

The steps for using ANNs to predict sensor data are as follows:

- Data preparation: This involves collecting and preprocessing the sensor data, including cleaning, normalization, and feature engineering.
- Splitting the data: The sensor data is divided into training and testing sets.
- Model training: The ANN model is trained on the training set using backpropagation, which updates the weights between the neurons to minimize the error between the predicted and actual output.
- Model evaluation: The model's performance is evaluated using the testing set, and metrics such as mean squared error (MSE) and R-squared are calculated.
- Model optimization: The model can be optimized by tuning the hyperparameters, such as the number of hidden layers, the number of neurons per layer, and the learning rate, [5].

ANNs offer several advantages for predicting sensor data, including their ability to model nonlinear relationships between input and output variables, their robustness to noise and missing data, and their adaptability to changing environments. Additionally, ANNs can be used for regression and classification problems, [7].

2.3.1 Methodology for Implementing

ANNs find frequent use in predicting environmental information, such as air quality, [13], and water quality, [12], using sensor data.

Selecting the relevant independent and dependent variables based on environmental sensor data is the initial step in implementing ANNs. A training set of data is utilized to create and train the ANN model, requiring adjustments to the neural network's weights and biases to reduce the discrepancy between actual and predicted values of the dependent variable. Utilizing backpropagation, the algorithm updates the weights and biases of the neural network in response to the error gradient during the training process. A distinct test set of data is assessed to examine the ANN model's accuracy and precision in anticipating the dependent variable.

2.3.2 Performance of the Algorithm in Terms of Accuracy and Precision

Using metrics like the R-squared (R^2) coefficient and mean absolute error (MAE), the effectiveness of the ANN algorithm is assessed, [7]. The accuracy and precision of this algorithm are contingent on the quality and quantity of environmental sensor data, in addition to proper independent variable selection, [7], [12]. When there are nonlinear connections between independent and dependent variables, ANNs can outperform the Random Forest and linear regression algorithms, [7], [9]. Mean squared error (MSE) is another metric used to evaluate the performance of the ANN algorithm.

2.3.3 Limitations and Challenges

ANNs, while a productive machine learning algorithm, come with certain limitations and obstacles. Despite their strength, they can lead to a lack of generalizability when overfitted to the training data. They also require much training data and computational resources, limiting their applicability, [9]. Lastly, a daunting task lies in expertly selecting architecture and hyperparameters for the ANN model through experimentation, [5].

2.4 Support Vector Machines

Support Vector Machines (SVMs) are a popular machine learning model for predicting sensor data, capable of handling regression and classification problems, [9]. SVMs work by identifying a hyperplane in the input space that separates the data into two classes for classification or that best fits the data for regression. This hyperplane is chosen to maximize the margin between the two classes or the distance between the hyperplane and the data points, [11].

SVMs are particularly useful for handling high-dimensional data, including continuous and categorical input variables. The general steps for using SVMs to predict sensor data involve data preparation, splitting the data into a training and testing set, model training, evaluation, and optimization, [3]. During model training, the SVM model finds the hyperplane that best separates the data into two classes or best fits the data. The model's performance is evaluated using mean squared error (MSE) and R-squared, [11].

SVMs have several advantages for predicting sensor data, including high accuracy due to the optimal hyperplane and the ability to handle linear and nonlinear relationships between input variables and output variables using different kernel functions, [9], [11]. Additionally, SVMs are robust to overfitting, which can occur when a model is too complex and captures noise in the data. SVMs are also versatile, making them popular for various machine learning applications, [13].

2.4.1 Methodology for Implementing

Predictive algorithms known as Support Vector Machines (SVMs) have their place in classification and regression. Often, they are implemented in determining environmental factors, like air and water quality, concerning sensor data, [14]. Contrastingly, they first choose which parameters and environmental indicators are relevant before continuing by using a kernel function that raises the information to a higher level of complexity. The algorithm then proceeds by using a linear hyperplane to differentiate the classes. Training the SVM model on a data set maximizes the margin between the classes. Then, the SVM model's accuracy and precision in predicting the dependent variable are evaluated using a separate test set of data, [15].

2.4.2 Performance of the Algorithm in Terms of Accuracy and Precision

Various metrics are available to evaluate the SVM algorithm's performance, including mean absolute error (MAE), mean squared error (MSE), and R-squared (R^2) coefficient, [9]. The performance of the SVM algorithm depends on the proper selection of the kernel function and the quality and quantity of environmental sensor data, [9]. When working with high-dimensional data or nonlinear relationships between dependent and independent variables, SVMs can outperform linear regression and Random Forest, [2].

2.4.3 Limitations and Challenges

For SVMs, there are a few hindrances and obstacles to be aware of. One issue is that their performance can be affected by the selection of kernel functions and hyperparameters – this is something to keep in mind. Training and optimizing SVMs can also be computationally demanding and may hinder their scalability in certain situations. Lastly, grappling with interpreting the SVM model is a task because it relies on a multifaceted objective function, [9].

2.5 SARIMA

2.5.1 Methodology for Implementing

The order of seasonal differencing (D) and seasonal orders for autoregressive (p), integrated (q), and moving average (P and Q) components must be selected after identifying the seasonal pattern in the data when implementing SARIMA. The augmented Dickey-Fuller test, [6], and visual inspection are methods for choosing appropriate values.

Using maximum likelihood estimation, one must fit the SARIMA model to the data, but before that, one needs to establish the values for D, p, q, P, and Q. The idea is to determine the model parameters that would optimize the likelihood of observing the data based on the values established, [12].

2.5.2 Performance of the Algorithm in Terms of Accuracy and Precision

The accuracy and precision of the SARIMA algorithm are contingent upon both the quality of data and its intended use. A strong seasonal pattern with stationary data post-seasonal differencing provides the ideal circumstances for SARIMA to perform well – especially regarding seasonal time series data predictions. For data lacking a well-defined seasonal pattern, SARIMA was deemed unsuitable according to a study of various time series forecasting approaches. In cases where data displayed a distinct monthly trend, SARIMA surpasses other techniques.

2.5.3 Limitations and Challenges

Challenges and limitations are present in this scenario, causing difficulty in achieving objectives. In the realm of time series forecasting, SARIMA comes with its fair share of challenges and limitations, [17]. One of its challenges involves the data's stationarity, which must be achieved via seasonal differencing. However, this task can prove to be difficult, especially if the data isn't stationary to begin with.

In practice, a linear process generating the data is not always assumed by SARIMA, presenting another challenge. It is also sensitive and may not function effectively when the data contains significant anomalies or outliers.

Interpreting the model parameters and grasping the dynamics behind time series data is a complicated feat with SARIMA, which is a notable drawback. This poses a challenge when trying to utilize SARIMA for causal inference or to seek comprehension of the underlying mechanisms producing the data. Ultimately, SARIMA falls short in providing easy-to-understand explanations or insights.

2.6 Hybrid Approach

In machine learning, a hybrid model for predicting sensor data involves combining multiple models to enhance predictions' accuracy or address individual models' limitations. For instance, one approach for predicting temperature sensor data is using a hybrid model that combines a linear regression model and a random forest model.

The sensor data is first prepared by cleaning, normalizing, and engineering features to create this hybrid model. The data is then split into training and testing sets, and two models are trained on the training set: a linear regression model that uses previous temperature readings to predict the next reading and a random forest model that incorporates additional features like time of day, day of the week, and season.

The predicted values from both models are then combined using a weighted average, with weights determined based on each model's performance on the training set. Finally, the performance of the hybrid model is evaluated using the testing set, with metrics like mean squared error and R-squared calculated.

Another example of a hybrid approach for predicting sensor data is the combination of Random Forest and Artificial Neural Network models for predicting air quality using environmental sensor data, [13]. The Random Forest model was used to select essential features from the sensor data, which were then used as inputs to the Artificial Neural Network model for air quality prediction. The hybrid approach outperformed both individual models in terms of prediction accuracy.

The advantages of using a hybrid model for sensor data prediction include the ability to capture both linear and nonlinear relationships between input and output variables, as well as the ability to incorporate additional features that can improve prediction accuracy. Moreover, combining multiple models can help reduce the risk of overfitting and improve prediction robustness. However, the specific hybrid model that is most appropriate will depend on the nature of the data and the problem being addressed.

2.6.1 Methodology for Implementing

To enhance the precision and accuracy of environmental sensor data forecasts, a hybrid strategy fuses numerous machine learning algorithms, including artificial neural networks, support vector machines, Random Forest, and linear regression. A hybrid approach selection process requires evaluating the specific combination of algorithms best suited for the type of data and prediction issue at hand. A widely adopted hybrid approach merges the strengths of Random Forest and linear regression for improved efficacy. To assess the accuracy and precision of the hybrid model's prediction of the dependent variable, it must first undergo training with a data set and then evaluation through an alternative test set.

2.6.2 Performance of the Algorithm in Terms of Accuracy and Precision

By considering the strengths of different algorithms, hybrid models can achieve greater accuracy and precision compared to individual algorithms, which can be assessed through various metrics such as mean absolute error (MAE), mean squared error (MSE), and the R-squared (R^2) coefficient. The success of the hybrid approach is primarily determined by the quality and quantity of the environmental sensor data, the relevance of the chosen algorithms for the hybrid model, and the techniques utilized to integrate the results of various algorithms.

2.6.3 Limitations and Challenges

Optimal algorithm combinations can be complicated when implementing a hybrid approach for forecasting. The cost of computing during hybrid model training and optimization can also prove pricier than with solo algorithms. Furthermore, hybrid models can intensify model complexity, thus rendering result interpretation and revision a bit trickier.

3 Comparison of Methods

This study compared five methods for forecasting sensor data: random forest, artificial neural networks, convolutional neural networks, and autoregression. Each method has its strengths and weaknesses, which we summarize below:

Random forest is an ensemble method that combines multiple decision trees to improve accuracy and reduce overfitting. It can handle continuous and categorical input variables and capture nonlinear relationships between input and output variables, [18]. However, training can be slow and requires more computational resources.

Artificial neural networks can handle complex nonlinear relationships between input and output variables and can handle both continuous and categorical input variables. However, they can be prone to overfitting if the model is too complex and requires a large amount of data for training. Artificial neural networks can also be trained slowly and require more computational resources.

Convolutional Neural Networks (CNNs) are highly effective for tasks involving spatial patterns and hierarchical features in data, such as image and time-series analysis. They can handle both continuous and categorical input variables and manage complex relationships between inputs and outputs. CNNs consist of convolutional layers that detect local patterns, pooling layers that reduce spatial dimensions, and fully connected layers that integrate learned features. These networks are adept at capturing intricate structures in data and can generalize well, although careful architecture design and hyperparameter tuning are required to optimize performance and prevent overfitting.

Ultimately, the choice of method for predicting sensor data will depend on the nature of the data and the specific problem at hand. For simple problems with linear relationships, linear regression may be a good choice, while more complex issues may require methods like random forests or artificial neural networks. Therefore, this study aims to experiment with multiple methods and compare their performance to choose the best one for sensor data forecast.

3.1 Model Configurations

Hyperparameters for the developed models are chosen based on hyperparameter optimization techniques, including random search, [19], [20], and grid search, [21], [22], to identify the optimal parameter set which gives the highest accuracy. The parameters and their boundaries are described as follows.

3.1.1 LSTM Model

In the given example, LSTM hyperparameter optimization was performed using a random search approach with the Keras Tuner. The objective was to minimize the loss while searching for the best combination of hyperparameters.

The hyperparameters that were optimized, and their respective search ranges include:

- units_1: The number of units in the first LSTM layer. It was tested with values ranging from 30 to 100 with a step of 10.
- units_2: The number of units in the second LSTM layer. It was tested with values ranging from 30 to 100 with a step of 10.
- max_trials: The maximum number of hyperparameter combinations to try. It was set to 5.
- executions_per_trial: The number of times to execute each trial with the same hyperparameters. It was set to 2.

The LSTM model was built with two LSTM layers followed by a dense output layer. The model was compiled using the Adam optimizer and the mean squared error (MSE) loss function. The random search explored various combinations of hyperparameters within the specified ranges, evaluating the performance of each combination using a 20% validation split on the training data.

3.1.2 VAR Model

For VAR model implementation, the primary hyperparameter to optimize is the number of lags (p) included in the model. The selection of the optimal lag order can significantly impact the model's forecasting performance. Selecting the optimal lag order based on an information criterion like AIC (Akaike's Information Criterion), [23]. For the given dataset, the AIC value was identified as 15.

3.1.3 CNN Model

The CNN hyperparameter optimization was carried out using a grid search approach. The "Keras Regressor" was utilized to create a compatible model for a grid search. The model's parameters are explored within a specified range, and the optimal combination of these parameters is determined to provide the best performance, [24].

Here are the hyperparameters being optimized and their respective search ranges:

- filters: The number of filters in the convolutional layer. It was tested with 16, 32, and 64 filters.
- kernel_size: The size of the convolutional kernel (window). It was tested with kernel sizes of 2, 3, and 4.
- pool_size: The size of the pooling window in the max-pooling layer. It was tested with pool sizes of 1, 2, and 3.
- dense_units: The number of units in the dense (fully connected) layer. It was tested with 30, 50, and 100 units.

The grid search method performed a search over the specified parameter ranges, evaluating the performance of each combination.

3.1.4 ANN Model

Artificial Neural Network (ANN) hyperparameter optimization was performed using a random search approach with the "Keras Tuner". The objective was to minimize the validation loss while searching for the best combination of hyperparameters:

- units_input: The number of units in the input layer. It was tested with values ranging from 32 to 512 with a step of 32.
- units_hidden: The number of units in the hidden layer. It was tested with values ranging from 32 to 512 with a step of 32.
- learning_rate: The learning rate for the Adam optimizer. It was tested with values of 0.01, 0.001, and 0.0001.

The random search was configured with the following settings:

- max_trials: The maximum number of hyperparameter combinations to try. It was set to 5.
- executions_per_trial: The number of times to execute each trial with the same hyperparameters. It was set to 3.

The random search explored various combinations of hyperparameters within the specified ranges, evaluating the performance of each combination. The best combination of hyperparameters was selected based on the lowest validation loss.

3.1.5 Random Forest Model

Random forest model hyperparameter optimization was carried out by using a grid search approach by experimenting with different numbers of decision trees. The hyperparameters that were optimized, and their respective search ranges include:

- n_estimators: The number of decision trees in the forest. It was tested on 10, 50, 100, and 200 trees. It was observed that there was no significant decrease in the prediction error with more decision trees beyond the tested values, which informed the decision to limit the number of decision trees in the Random Forest model to these ranges.
- max_depth: The maximum depth of each decision tree. It was tested with no limit (None), 10, 20, and 30 levels.
- min_samples_split: The minimum number of samples required to split an internal node. It was tested with values of 2, 5, and 10.
- min_samples_leaf: The minimum number of samples required to be at a leaf node. It was tested with values of 1, 2, and 4.
- max_features: The number of features to consider when looking for the best split. It was tested with 'auto' (equivalent to 'sqrt') and 'sqrt' options.

The models were trained and evaluated with each of these configurations to determine the optimal values of parameters that would yield the best performance in terms of error reduction, [22].

For predicting environmental sensor data, selecting the optimal number of layers in a neural network and the number of trees in a random forest model is dependent on different variables, like the size of the dataset, the available computation resources, the complexity of the problem, and the quality of the data. The below points are some general rules to keep in mind.

Models employing the random forest approach have the following characteristics:

For determining how many trees are optimal for a given problem and dataset, cross-validation techniques like k-fold cross-validation come in handy, [25]. Based on the amount of data available and the complexity of the problem, it is possible to determine the number of layers to use.

While a small dataset with rudimentary designs may only require one or two hidden layers, a more extensive dataset with complicated patterns might demand more hidden layers. If one were to make the model more intricate, it might increase one's capacity to grasp ambiguous patterns. Still, if the model becomes overly complicated, it could also heighten the chance of overfitting.

A given problem and the dataset's ideal layers can be found using early stopping and cross-validation techniques, [25].

For optimal performance, it's crucial to properly tune the hyperparameters of the models, like learning rate, regularization, and activation functions, among other things.

4 Data Processing and Experimentation

The accuracy of the sensor forecasting data was evaluated by comparing the error between actual sensor readings and forecasted sensor values. The experiment included comparing forecast accuracy with models such as Random Forest, Convolutional Neural Networks (CNNs), Artificial Neural Networks (ANN), Vector Autoregressive (VAR) model forecast, and Long Short-Term Memory

(LSTM) networks. Several models were developed to compare the forecast accuracy and performance (Fig. 1).

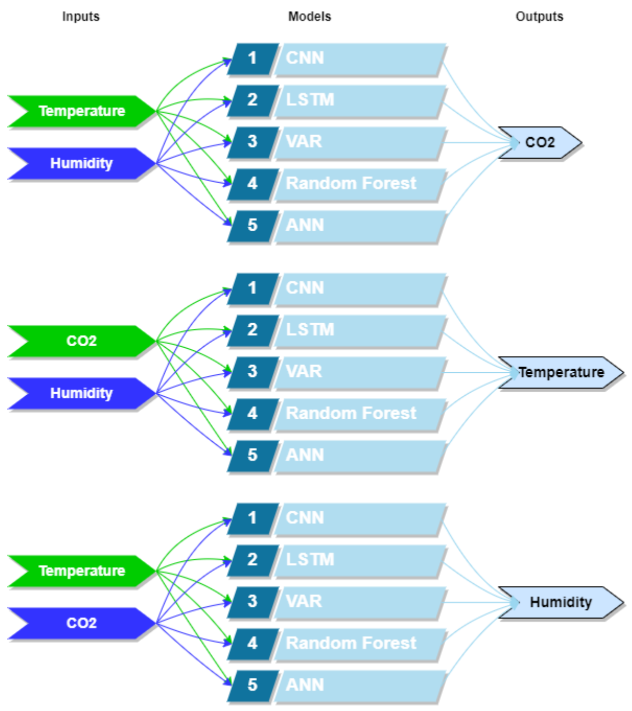

Fig. 1: The implemented sensor data forecasting models

Each model has 2 inputs or sensor parameters, such as Humidity and CO_2, and the output of the models is a forecast, for example, temperature.

4.1 Experiment Preparation

Handling missing values and normalizing the temperature, humidity, and CO_2 readings were initial preprocessing steps for raw sensor data. Additionally, the time-series data was transformed into a proper format for model input. As a result, sensor measurements are converted into individual variables in a multivariate time-series organization.

A basic model was developed using CNN, consisting of a single convolutional layer followed by several pooling layers, and then completed with fully connected layers. The convolutional layers are designed to identify local patterns within the input data, while the pooling layers serve to decrease spatial dimensions and reduce computational complexity. The highly established layers remap the extracted distinctive attributes to dictate the conclusive prediction. Concerning the time-series information, the model applies 1D convolutions instead of the standard 2D convolutions in image processing. Sensor measurements are subjected to local pattern recognition by sliding convolutions over input data. Filters are utilized to capture multiple patterns, followed by merging resultant outputs to form feature maps. ReLU (Rectified Linear Unit) activation is applied to post-convolutional and fully connected layers to introduce nonlinearity, which enables the model to learn intricate relationships and patterns, [26], [27], [28].

The CNN and LSTM models were rigorously evaluated, with the forecast and actual sensor readings being compared using the Mean Squared Error (MSE) as the loss metric. To adjust the network's weights during training, "Adam," an optimization algorithm, was called to minimize the loss function. The final predictions are mapped via the fully connected layer following two LSTM layers in the LSTM network model. Temporal dependencies are captured by the LSTM layers while extracting features.

Recognition of local patterns in the time-series data was possible using 1D convolutions. At the same time, introducing the ReLU activation function was integral in enabling the learning of complex patterns with nonlinearity. The LSTM model utilized the dense layer to generate final predictions from the data with captured temporal dependencies. For the VAR model, coefficients were estimated to forecast upcoming sensor measurements after data preprocessing and determining the lag order using AIC. Measuring the distinction between forecasted and actual measurements was done with the loss function mean squared error. The "Adam" optimization algorithm was implemented to minimize the loss function.

To use the VAR model, missing values must be taken care of, followed by a check for stationarity. In the event of non-stationarity, different transformations or differencing must be applied.

The process of selecting the ideal lag order for the VAR model, which determines how many prior time steps should be considered when forecasting future values, is known as model selection. The Akaike Information Criterion (AIC) is utilized for this purpose. After confirming that the residuals are acceptable, the VAR model can be utilized to anticipate future sensor readings. The forecasts are generated using the past temperature, humidity, and CO_2 data values in conjunction with the estimated coefficients, [29].

4.2 Running Experiments

Having conducted several trials on each model, 15 sensor forecast models were compared based on their ability to forecast the latest 100 entries in the dataset. The most accurate results for LSTM and CNN were obtained after 10 epochs. To ensure the credibility of the results, each model underwent training at least 30 times, and the outcome was determined by calculating the average error value. The processing time of each model was also recorded using the same methodology. The models were trained using a hardware configuration of 20GB RAM and a 1.60GHz Intel Core i5-8250U CPU.

4 Results

To evaluate the performance of developed forecast models, Mean Squared Error (MSE) and Mean Absolute Error (MAE) were used (Table 1.). Also, the training time was noted when training different models.

Table 1. Comparison Of Forecast Errors

		Forecasted sensor		
		Temperature (humidity and CO_2 as inputs)	**Humidity (temperature and CO_2 as inputs)**	**CO_2 (humidity and temperature as inputs)**
Model	**LSTM**	MAE: 0.0696, MSE: 0.00712, Time: 134.88 s	MAE: 1.019, MSE: 1.516, Time: 108.65 s	MAE: 95.952, MSE: 15529.301, Time: 109.17 s
	VAR	MAE: 0.0696, MSE: 0.00675, Time:0.69s	MAE: 27.074, MSE: 733.033, Time:0.70 s	MAE: 1424.80, MSE: 2030813.5, Time: 0.70s
	CNN	MAE: 0.147, MSE: 0.0289, Time: 11.256 s	MAE: 0.533, MSE: 0.4006, Time: 11.377 s	MAE: 264.951, MSE: 86007.97, Time: 11.301 s
	ANN	MAE: 0.378, MSE: 0.293, Time: 25.869 s	MAE: 3.30, MSE: 17.87, Time: 25.494 s	MAE: 184.35, MSE: 61684.87, Time: 26.01s
	Random Forest	MAE: 0.1607, MSE: 0.132, Time: 3.989 s	MAE: 1.178, MSE: 5.4687, Time: 3.841 s	MAE: 32.582, MSE: 6651.02, Time: 2.803 s

The Mean Absolute Error result for all models is provided in "Fig. 2".

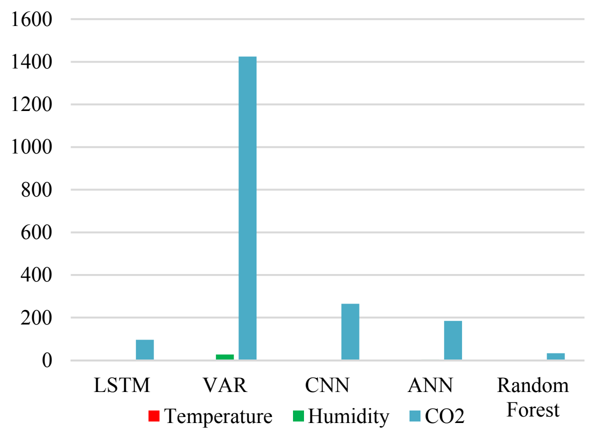

Fig. 2: MAE indicators

The Mean Squared Error results for all models are provided in "Fig. 3".

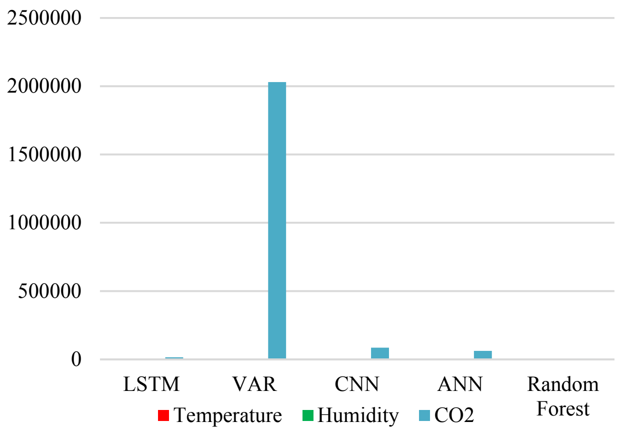

Fig. 3: MSE indicators

Model performance result for all models is provided in "Fig. 4".

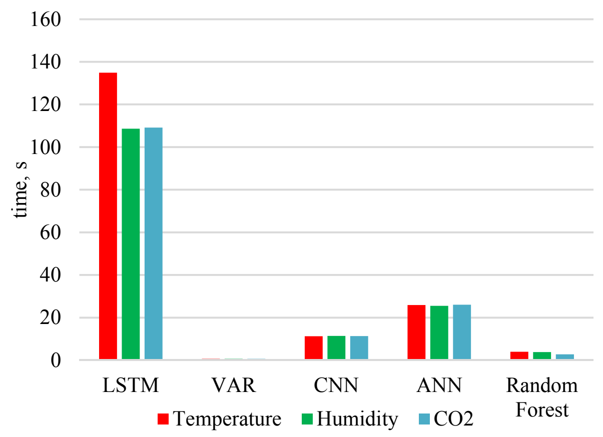

Fig. 4: Time indicators

Based on the results in the chart provided, we can gain a few insights about the ability of various machine learning models to predict environmental factors such as CO_2 levels, humidity, and temperature. In terms of Mean Absolute Error (MAE) and Mean Squared Error (MSE), it is generally observed that LSTM and VAR models are superior to the other models across all three categories. CO_2 levels were found to have the lowest value on the VAR model. In contrast, the LSTM model recorded the lowest MAE and MSE values for humidity and temperature. ANN and CNN models poorly forecast CO_2 levels, while temperature and humidity perform moderately. Mostly, the Random Forest model forecasts the temperature and humidity with high accuracy, although not when it comes to CO_2 forecasts. Other models have performed better in this regard.

5 Conclusion

Multiple techniques for predicting time series exist depending on the data and problem. Some popular options include Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, Vector Autoregressive (VAR) models, and Random Forests. Particularly useful with large datasets, the CNN method excels at recognizing localized patterns and acquiring knowledge of hierarchical feature representations, thus being highly efficient in computation. Despite these benefits, it may not be well-suited to handling distant dependencies in time series data, and one must be cautious of overfitting if proper regularization is not implemented.

Training the LSTM model could be computationally expensive, requiring additional time and resources, particularly for extensive input sequences or large datasets. Nonetheless, the LSTM is crucially outfitted to manage time series data with long-range dependencies and can memorize and learn patterns over extended durations, providing the most precise outcomes. When forecasting temperature from humidity and CO_2 data, LSTM can be used to achieve the highest accuracy if the training time is not a constraint.

Though the VAR model can efficiently implement and catch linear relationships among numerous time series, its dependency on stationarity and assumption of linearity may need to be revised for complex or nonlinear situations. While it functions optimally during constant correlation over time, problems surface when handling vast quantities of data or high-dimensional inputs.

The results reveal that the environmental sensor data can be forecasted using VAR or LSTM models. Across all three temperature, humidity, and CO_2 levels categories, these models outperformed the others in both mean absolute and squared errors. For temperature and humidity predictions, the LSTM model proves most effective. The least error was achieved when forecasting temperature from CO_2 and humidity inputs, thereby potentially serving as substitutes for physical temperature sensor devices. When it comes to CO_2, however, the forecast accuracy was low in comparison to other parameters.

Among the models tested, CNN and ANN show acceptable results regarding temperature and humidity, but they fared poorly regarding CO_2 levels. Although Random Forest performed well for temperature and humidity, its CO_2 forecast accuracy was less accurate than the other models. Regarding computation time, VAR and Random Forest stood out as the quickest, while LSTM and ANN proved to be the slowest.

References:

[1] Luis M. Candanedo, Véronique Feldheim. Accurate occupancy detection of an office room from light, temperature, humidity, and CO2 measurements using statistical learning models. Energy and Buildings. Volume 112, 15 January 2016, Pages 28-39.

[2] Méndez, M., Merayo, M.G. & Núñez, M. Machine learning algorithms to forecast air quality: a survey. Artif Intell Rev (2023). https://doi.org/10.1007/s10462-023-10424-4

[3] Kumar, K., Pande, B.P. Air pollution prediction with machine learning: a case study of Indian cities. Int. J. Environ. Sci. Technol. 20, 5333–5348 (2023). https://doi.org/10.1007/s13762-022-04241-5

[4] Yu, R., Yang, Y., Yang, L., Han, G., & Move, O. A. (2016). RAQ-A Random Forest Approach for Predicting Air Quality in Urban Sensing Systems. Sensors (Basel, Switzerland), 16(1), 86. https://doi.org/10.3390/s16010086

[5] Yuanlin Gu, Baihua Li, Qinggang Meng, Hybrid interpretable predictive machine learning model for air pollution prediction, Neurocomputing, Volume 468, 2022, Pages 123-136, ISSN 0925-2312, https://doi.org/10.1016/j.neucom.2021.09.051

[6] W.-I. Lai, Y.-Y. Chen, and J.-H. Sun, "Ensemble Machine Learning Model for Accurate Air Pollution Detection Using Commercial Gas Sensors," Sensors, vol. 22, no. 12, p. 4393, Jun. 2022, doi: 10.3390/s22124393. Available: http://dx.doi.org/10.3390/s22124393

[7] Shivang Agarwal, Sumit Sharma, Suresh R., Md H. Rahman, Stijn Vranckx, Bino Maiheu, Lisa Blyth, Stijn Janssen, Prashant Gargava, V.K. Shukla, Sakshi Batra, Air quality forecasting using artificial neural networks with real time dynamic error correction in highly polluted regions, Science of The Total Environment, Volume 735, 2020, 139454, ISSN 0048-9697, https://doi.org/10.1016/j.scitotenv.2020.139454

[8] Mengyuan Zhu, Jiawei Wang, Xiao Yang, Yu Zhang, Linyu Zhang, Hongqiang Ren, Bing Wu, Lin Ye, A review of the application of machine learning in water quality evaluation, Eco-Environment & Health, Volume 1, Issue 2, 2022, Pages 107-116, ISSN 2772-9850, https://doi.org/10.1016/j.eehl.2022.06.001.

[9] Kempelis, A., Romanovs, A. & Patlins, A. 2021, "Implementation of Machine Learning based Approach in IoT Network Prototype", Proceedings of the 9th IEEE Workshop on Advances in Information, Electronic and Electrical Engineering, AIEEE 2021.

[10] Yue Hu, Xiaoxia Chen, Hanzhong Xia, A hybrid prediction model of air quality for sparse station based on spatio-temporal feature extraction, Atmospheric Pollution Research, Volume 14, Issue 6, 2023, 101765, ISSN 1309-1042, https://doi.org/10.1016/j.apr.2023.101765.

[11] W.C. Leong, R.O. Kelani, Z. Ahmad, Prediction of air pollution index (API) using support vector machine (SVM), Journal of Environmental Chemical Engineering, Volume 8, Issue 3, 2020, 103208, ISSN 2213-3437, https://doi.org/10.1016/j.jece.2019.103208.

[12] Chen, Yingyi & Song, Lihua & Liu, Yeqi & Yang, Ling & Li, Daoliang. (2020). A Review of the Artificial Neural Network Models for Water Quality Prediction. Applied Sciences. 10. 5776. 10.3390/app10175776.

[13] Nilesh N. Maltare, Safvan Vahora, Air Quality Index prediction using machine learning for Ahmedabad city, Digital Chemical Engineering, Volume 7, 2023, 100093, ISSN 2772-5081, https://doi.org/10.1016/j.dche.2023.100093.

[14] Yang, H., Zhao, J. & Li, G. A new hybrid prediction model of PM2.5 concentration based on secondary decomposition and optimized extreme learning machine. Environ Sci Pollut Res 29, 67214–67241 (2022). https://doi.org/10.1007/s11356-022-20375-y.

[15] B. D. Parameshachari, G. M. Siddesh, V. Sridhar, M. Latha, K. N. A. Sattar and G. Manjula., "Prediction and Analysis of Air Quality Index using Machine Learning Algorithms," 2022 IEEE International Conference on Data Science and Information System (ICDSIS), Hassan, India, 2022, pp. 1-5, doi: 10.1109/ICDSIS55133.2022.9915802.

[16] Narigina, M., Osadcijs, E., & Romanovs, A. (2022). Analysis of Medical Data Processing Technologies. In 63rd International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS) (pp. 1-6). Riga, Latvia: IEEE. doi: 10.1109/ITMS56974.2022.9937120.

[17] Liu, X., Lin, Z., & Feng, Z. (2021). Short-term offshore wind speed forecast by seasonal ARIMA - A comparison against GRU and LSTM. Energy, 227, 120492. doi.org/10.1016/j.energy.2021.120492.

[18] Mehmood, K., Bao, Y., Saifullah, Cheng, W., Khan, M. A., Siddique, N., Abrar, M. M., Soban, A., Fahad, S., & Naidu, R. (2021). Predicting the quality of air with machine learning approaches: Current research priorities and future perspectives. Journal of Environmental Management. doi.org/10.1016/j.jclepro.2022.134656.

[19] Buslim, N., Rahmatullah I. L., Setyawan B. A. and Alamsyah A., "Comparing Bitcoin's Prediction Model Using GRU, RNN, and LSTM by Hyperparameter Optimization Grid Search and Random Search," 2021 9th International Conference on Cyber and IT Service Management (CITSM), Bengkulu,

Indonesia, 2021, pp. 1-6, doi: 10.1109/CITSM52892.2021.9588947.

[20] Vrskova R., Sykora P., Kamencay P., Hudec R. and Radil R., "Hyperparameter Tuning of ConvLSTM Network Models," 2021 44th International Conference on Telecommunications and Signal Processing (TSP), Brno, Czech Republic, 2021, pp. 15-18, doi: 10.1109/TSP52935.2021.9522683.

[21] Chandok A., Verma A. and Gupta R., "Dro-Mal Detector: A Novel Method of Android Malware Detection," 2022 3rd International Conference for Emerging Technology (INCET), Belgaum, India, 2022, pp. 1-9, doi: 10.1109/INCET54531.2022.9824877

[22] R. Vrskova, P. Sykora, P. Kamencay, R. Hudec and R. Radil, "Hyperparameter Tuning of ConvLSTM Network Models," 2021 44th International Conference on Telecommunications and Signal Processing (TSP), Brno, Czech Republic, 2021, pp. 15-18, doi: 10.1109/TSP52935.2021.9522683.

[23] Portet S., A primer on model selection using the Akaike Information Criterion, Infectious Disease Modelling, Volume 5, 2020, Pages 111-128, ISSN 2468-0427, doi.org/10.1016/j.idm.2019.12.010.

[24] Zhang, W., Zhou, H., Bao, X., & Cui, H. (2023). Outlet water temperature prediction of energy pile based on spatial-temporal feature extraction through CNN–LSTM hybrid model. Energy, 264, 126190. doi:10.1016/j.energy.2022.126190.

[25] H. K. Skrodelis and A. Romanovs, "Synthetic Network Traffic Generation in IoT Supply Chain Environment," 2022 63rd International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS), Riga, Latvia, 2022, pp. 1-5, doi: 10.1109/ITMS56974.2022.9937126.

[26] B. Y, S. D. B, S. V, A. D and E. S. R. M., "A Deep Learning Approach to Predict COVID-19 Through Cough Analysis Using CNN-BiDirectional LSTM," 2022 International Conference for Advancement in Technology (ICONAT), Goa, India, 2022, pp. 1-5, doi: 10.1109/ICONAT53423.2022.9726067.

[27] A. Kempelis, A. Romanovs and A. Patlins, "Design and Implementation of IoT Network Prototype to Facilitate the Food Production Process in Agriculture," IEEE EUROCON 2021 - 19th International Conference on Smart Technologies, Lviv, Ukraine, 2021, pp. 71-76, doi: 10.1109/EUROCON52738.2021.9535556.

[28] A. Kempelis, A. Romanovs and A. Patlins, "Using Computer Vision and Machine Learning Based Methods for Plant Monitoring in Agriculture: A Systematic Literature Review," 2022 63rd International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS), Riga, Latvia, 2022, pp. 1-6, doi: 10.1109/ITMS56974.2022.9937119.

[29] M. Narigina, A. Kempelis, A. Romanovs, E. Osadcijs and A. Patlins, " Machine Learning based Sensor Data Forecasting for Precision Evaluation of Environmental Sensing" 2023 The 10th Jubilee IEEE Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), Vilnius, Lithuania, 2023, pp.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed to the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

The research has been supported by the RTU internal project competition for strengthening the capacity of scientific staff.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.