applied sciences

MDPI

Article

Vibration Velocity Prediction with Regression and Forecasting Techniques for Axial Piston Pump

Paweł Fic [1,*], Adam Czornik [1] and Piotr Rosikowski [2]

[1] Department of Automatic Control and Robotics, Silesian University of Technology, 44-100 Gliwice, Poland; adam.czornik@polsl.pl
[2] PONAR Wadowice S.A., Św. Jana Pawła II 10, 43-170 Łaziska Górne, Poland; piotr.rosikowski@ponar-wadowice.pl
* Correspondence: pawel.fic@interia.com; Tel.: +48-506-835-509

Abstract: Measuring vibration velocity is one of the most common techniques to estimate the condition of industrial machines. At a constant operating point, as the vibration velocity value increases, the machine's condition worsens. However, there are no precise thresholds that indicate the condition of a machine at different operating points. Also, the axial piston pump, which is the subject of the article, is a device that generates stronger vibrations by design and cannot be enclosed in general vibration norms. Due to different use cases and work regimes of axial piston pumps, the need to determine whether the device is working correctly for a broad spectra of operating points emerges. This article aims to present and compare different methods for vibration velocity prediction for axial piston pumps with use of neural networks including dense networks, variants of recurrent neural networks, and ensemble methods. The result of this research consists of models that have performance metrics that clearly indicate whether the monitored pump has malfunctioned or not across a wide variety of operating points, working conditions, and in case of reassembling. A detailed analysis of the influence of available measured variables on the performance of models is also provided. The conclusion is that the application of commercial implementation of developed models is reasonable in the context of both performance quality and costs of sensors needed to provide the necessary data.

Keywords: neural networks; virtual sensor; vibrations; sensor fusion; axial piston pump

check for updates

Citation: Fic, P.; Czornik, A.; Rosikowski, P. Vibration Velocity Prediction with Regression and Forecasting Techniques for Axial Piston Pump. *Appl. Sci.* **2023**, *13*, 11636. https://doi.org/10.3390/app132111636

Academic Editor: Jean-Jacques Sinou

Received: 20 September 2023
Revised: 13 October 2023
Accepted: 18 October 2023
Published: 24 October 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Industrial processes often operate in precisely described conditions. Any deviation from the operating point could lead to a reduction in production throughput. Additionally, industrial devices that are forced to work beyond the checked and validated operating point could result in failures. Therefore, tools that can prevent the device from operating under undesirable conditions are highly regarded [1]. One of the well-known methods to keep the condition of a machine in check is vibration velocity monitoring. The most common indicator used in industry is the root mean square value of vibration velocity (V_{RMS}) [2]. According to its value, the condition of the machine could be roughly estimated with the use of the ISO norm [3]. However, specified thresholds are not well-suited for all applications because of two main reasons:

- The ISO norm does not enclose a comprehensive analysis of machines that operates in different conditions (e.g., foundation, operating point);
- There are plenty of industrial device types, some of them could generate far stronger vibrations by design than ones specified in the norm.

In specific industrial devices machines' datasheets, some operating vibration velocity values are provided. However, they refer to limited working conditions and loads. Due

to different possible regimes of work and use cases for axial piston pumps, a tool for estimating the vibration velocity is desired.

The article presents several techniques for predicting future vibration velocity values. As a target, deviations from the nominal condition of the machine will be used to determine whether the monitored device has somehow malfunctioned. The approach presented in this article is correlated with the concept of a virtual sensor presented in article [4]. Such an approach, in different use cases and domains, was presented in publication [5].

In addition, ensemble learning [6] was used to mix different techniques together and train one strong-learner instead of several weak-learners. Such an ensemble approach can potentially give better results than using single models without blending them [7]. All described methods were evaluated and compared. The Python programming language with packages Tensorflow [8] and Keras [9] was used to perform analysis and neural network training, scikit-learn [10] for training machine learning models, numpy [11], Pandas [12] matplotlib [13] and seaborn [14] for general data analysis and model evaluation.

Piston pump domain experts from PONAR Wadowice S.A. were consulted while writing the present article. Measurement experiments also took place there.

1.1. Related Work

Common applications of neural networks in industry include projects related to the subject of intelligent machine maintenance [15]. The purpose of this kind of project is to track the condition of a machine, so that potential failure of the machine could be detected or even predicted in case of Predictive Maintenance [16] implementation. Such an approach allows manufacturers to keep production flow undisturbed as much as possible.

Data acquisition plays a crucial role in intelligent maintenance. Without the proper measurement of quantities related to monitored machine or process, it is impossible to implement intelligent maintenance strategies based on machine condition monitoring. Data from sensors is the main source of information regarding the state of a machine [17].

In simple maintenance strategies, tracking the value of a chosen variable and applying a threshold to it can provide enough information. In such a case, it may not be necessary to implement more complex strategies, as the costs of development of complex strategies could be too high to ensure a return of investment, considering potential profits from the reduction of disturbances in machine work [18]. An example of such an implementation in industry could be vibration monitoring of a suspension bridge, as presented in [19]. In the article, the authors set warning threshold of RMS of vibration acceleration at 5 $\frac{\text{cm}}{\text{s}^2}$.

In many industrial implementations, the application of a threshold on measurements may be not enough in processes that work at different operating points [20]. Significant process nonlinearities make handmade determination of threshold values related to specific operating points difficult—that is where neural networks come into play. By training a network using data from different machine conditions at various operating points, a neural network is able to "learn" the behavior of the piston pump, even if this behavior is strongly nonlinear [21].

In the literature, applications of using neural networks for regression and forecasting purposes in different industry domains are presented. As an example, neural networks models were used for forecasting electricity consumption in the Brazilian industrial sector [22]. The presented results show that neural network-based approaches outperform classical forecasting algorithms over a longer prediction horizon in terms of mean absolute percentage error (MAPE), which for a 24 h prediction horizon equals 3.42%. In the paper [23], the authors present the facilitation of a neural network for production schedule optimization.

For the purpose of the present article, axial piston pump vibration velocity monitoring and prediction is the particular subject of research. Previous research described in [21] presents failure analysis and intelligent identification of critical friction pairs of an axial piston pump. Diagnostics of the pump were implemented with the use of pressure signal analysis transformed using wavelet transform and convolutional neural networks (CNN).

The authors claim that the proposed approach achieved 100% accuracy for recognition of different faults. These faults include wear of the slipper, wear of the swash plate, failure of the loose slipper, wear of the central spring, and no visible failure.

Another application of neural networks in piston pump state recognition and failure prediction is presented in the paper [24]. By using a feed forward neural network, 92.8% accuracy was achieved for the recognition of six distinct failures. Statistical features extracted from vibration data were input to the network.

The prediction of vibration velocity of an axial piston pump was presented in [25]. However, instead of using neural networks to model the course of pump vibration, a set of equations was used and frequency of vibration was a particular interest. The presented approach leads to an 8.85% error in vibration frequency estimation. Similar work was carried out and presented in [26]. However, the authors investigated frequency calculated with mathematical models for an axial piston pump with swash plate fault.

The virtual sensors concept constitutes one of the main maintenance practices applied in the industry. It allows one to obtain the value of some quantity without measuring it directly, but with the use of other measurable variables [27]. For particular cases, a virtual sensor is designed to determine vibration-related quantities. As presented in [20], virtual vibration sensors could be implemented for the purpose of real-time monitoring of NO_x emissions of a diesel engine. In the aviation industry, the application of a virtual vibration sensor in the context of fatigue life estimation was shown in [28]. The prediction of vibration measured in a high-speed train was described in [29]. The authors used recurrent neural networks (RNN) and LSTM networks and achieved 99% prediction accuracy.

1.2. Business Needs

In hydraulic systems where it is desired to use pumps with a constant flow rate, simple gear pumps are most often used, the main limitation of which is the relatively average operating pressure of max. 250 bar. In systems where long-term operation under high load (high pressure) is required with an extensive work cycle, piston-axial pumps are most often used, which, thanks to various regulators, enable a smooth change in flow rate and, consequently, a change in the operating speed of the actuators or hydraulic motors. These types of pumps enable operation at pressures up to 380 or even 420 bar [30]. Examples of sectors in which these type of hydraulic systems with axial piston pumps are used are test stands in the automotive sector or specialized mobile machines (e.g., mining, construction machines or defense industry machines).

In the field of power hydraulics, many manufacturers have reached the peak of their product development capabilities due to certain limitations imposed by fluid mechanics, which is the basic field on which power hydraulics is based. Therefore, in order to be competitive in the hydraulics market, hydraulics manufacturers see great potential in offering classic hydraulic systems, but equipped with innovative expert systems that, thanks to the existing knowledge base, will be able to inform users about emergency states among pumps, which are the "heart" of every hydraulic system.Most often, an unplanned failure of the pump and its replacement is not a problem in terms of the costs associated with purchasing a new one, but downtime of the entire technological process, e.g., in the automotive or mining sector, is extremely expensive, because each hour of downtime generates costs that significantly exceed the hydraulic elements used [30]. Therefore, hydraulic manufacturers see predictive maintenance systems as having the potential to expand the market for the hydraulic systems they offer.

1.3. Article Contributions

The main contributions to the state-of-the-art so far can be detailed as follows:

1. The methods presented in the literature lack the presentation of vibration velocity prediction for axial piston pump in the context of the virtual sensor concept with the use of neural networks.

2. In the literature, there is no comprehensive comparison between several approaches involving neural networks, specifically for the purpose of vibration velocity prediction for axial piston pumps related to the virtual sensor concept.

Therefore, the contributions of the present paper are significant.

2. Materials and Methods

2.1. Measurement Experiments

The device under test was an axial piston pump model HSP-10V manufacured by Hydraut [31]. In total there were used two vibration sensors, three temperature sensors, three pressure sensors and one flow sensor. Figure 1 schematically illustrates the experimental setup of the axial piston pump with installed sensors. In Figure 2, a photo of the setup is presented. All of the measured variables were acquired every second. Vibration-related features, namely RMS and P2P, were calculated by a measurement module from a buffer of 4096 samples acquired with 20 kHz frequency. All the measured variables are listed in Table 1.

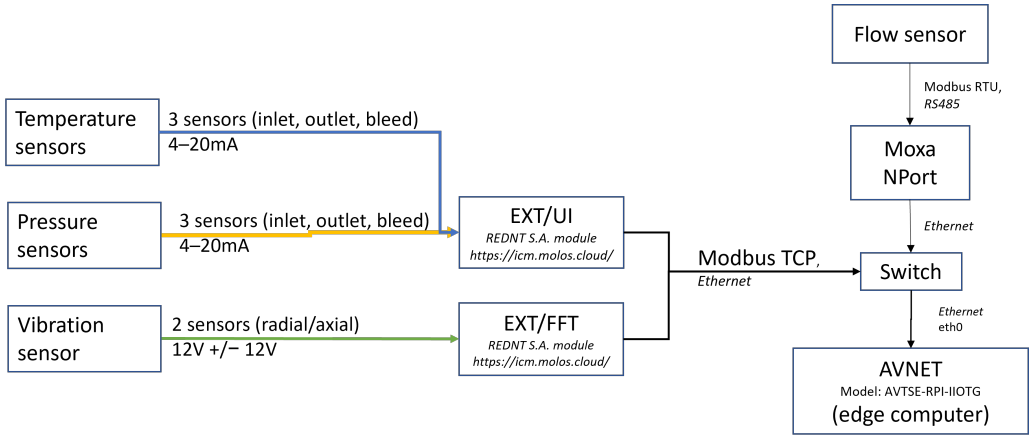


Figure 1. Experimental setup scheme of the axial piston pump with installed sensors. Colors in the scheme correspond with colors on Figure 2. REDNT S.A. modules can be found on the website https://icm.molos.cloud/ (accessed on 10 September 2023).

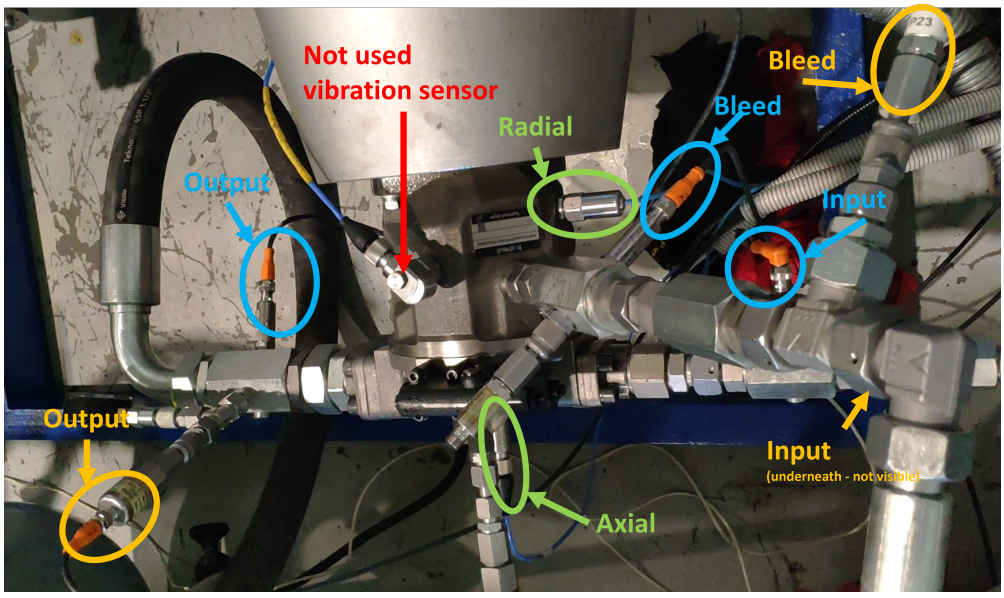


Figure 2. Experimental setup photo of the axial piston pump with installed sensors. Vibration sensors are marked green, pressure sensors are marked orange and temperature sensors are marked blue. These colors correspond with Figure 1.

The pump was tested in different conditions. Measurements were carried out on the pump that operated under the following experiments:

- Reference tests (twice)
- Reference tests with check valve
- Axial play fault
- Reassembling
- Timing plate fault
- Swash plate fault
- Bearing fault

Table 1. List of measured variables during experiments.

Vibration	Variable Name
Vibration velocity	Radial RMS/P2P, Axial RMS/P2P
Temperature	Inlet, Outlet, Bleed
Pressure	Inlet, Outlet, Bleed
Flow	Outlet

The same pump was used for all experiments. All of the failures were precisely performed by domain experts, so that specific failure could be isolated.

The set point for the pump during the tests was the torque load on the pump of the braking motor. Every experiment followed the same scenario as described in Table 2.

Table 2. Steps of measurement experiment. For constant load, period property is not applicable.

Step	Load	Period	Duration
Constant load	10, 20, . . . , 100%	x	3 min each
Trapezoid	30–90%	180 s	30 min
Sine	75–95%	60 s	15 min
Sine	10–100%	30 s	10 min

In the production environment there will be no measurable information about load, so it was not measured during the experiments. All of the developed approaches, model and techniques should therefore be designed in a way that does not involve information about the load.

Figure 3 presents the time series of vibration velocity measured by sensor mounted radial and output pressure. All of the experiments followed the same scenarios, but for "Reference tests v2" the load was lowered by mistake in the "Constant load" part for a couple of minutes.

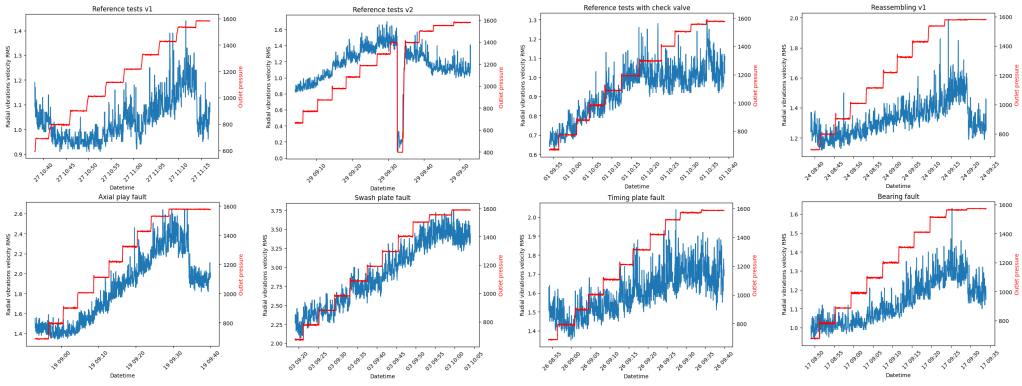


Figure 3. Visualization of all experiments. Blue line is $V_{RMS-Radial}$ (in $\frac{\text{mm}}{\text{s}}$) and red line is outlet pressure (in kPa).

2.2. Feature Importance Analysis

During the experiments, many different measurements were available. On the other hand, in the production environment, it will be difficult or sometimes impossible to use all variables indicated in Table 1. There are two main reasons for this. The first is the cost of sensors and second is the difficulty in mounting them properly on monitored device. Pumps are used in different conditions; in some use cases there is no space to fix a sensor on the device. For instance, in cases where other machines work nearby.

Domain experts from PONAR Wadowice suggested that sensors they consider the easiest to mount are sensors that measure outlet pressure, bleed pressure, outlet temperature and outlet flow. However, vibration sensor is the most desired factor because of its non-invasiveness and mobility. Considering possibilities of interference in the pump and advantages and disadvantages of mounting vibration sensors as presented in the article [32], sticking sensors with glue was performed on a test stand.

Two vibration sensors were available, however, in the production environment only one sensor is desired due to constraints and conditions described in the previous paragraph. To meet this requirement, an analysis of V_{RMS} value from both sensors for all carried out experiments for a constant load was performed. V_{RMS} values from only steady state periods were taken into consideration. Median values of analyzed variables were calculated. The results are shown in Figure 4.

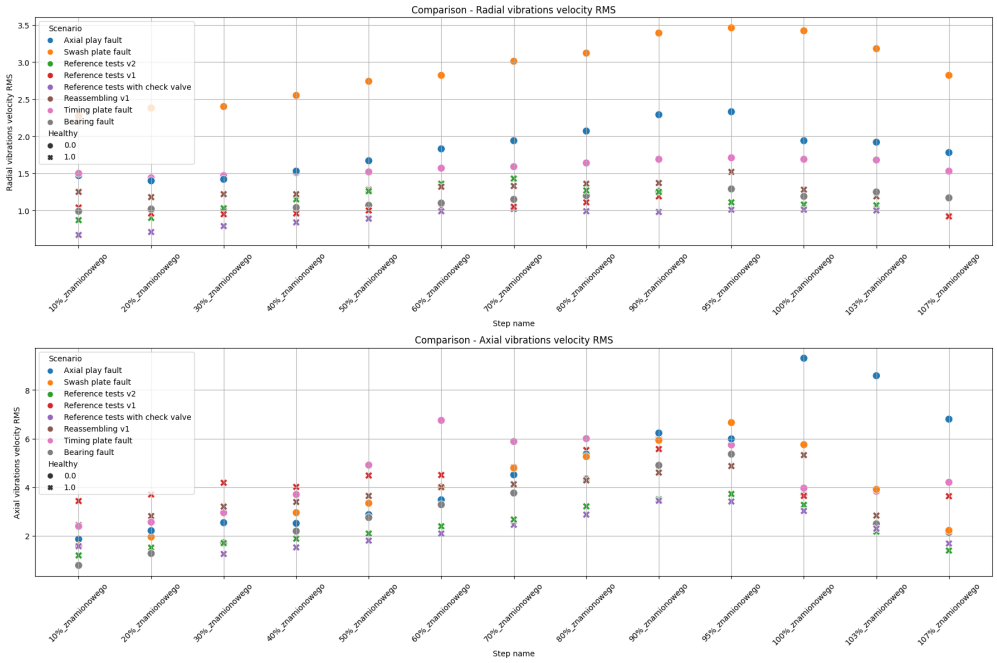


Figure 4. Median values for $V_{RMS-Radial}$ for each load value distinguishing between types of failures.

Based on analysis of Figure 4, several conclusions could be drawn. Measurements from a sensor mounted in radial position are more informative in the context of distinguishing between an undamaged and damaged pump. It is possible to set a threshold visually for every load that separates the pump with and without faults for all fault types, except bearing faults. However, this is not the case for every load or for every experiment, as the differences between V_{RMS} values are significant. On the other hand, for sensors mounted in axial position, such a simple threshold is not possible to establish. Taking the above factors into consideration, it could be deduced that measurements from sensors mounted in radial position are more informative for the description of pump condition. The goal of the article is therefore to estimate vibration velocity root-mean-square value that comes from a radial mounted sensor. This variable will be further denoted as $V_{RMS-Radial}$.

After selecting the most informative vibration sensor and choosing $V_{RMS-Radial}$ as the output variable of the model, input variables should be chosen. As presented in [33,34],

among all the available measured quantities, selecting the most informative ones to improve the quality of the deep learning model prediction could be performed using Pearson Correlation Coefficients (PCC). The higher the PCC absolute value between the output and the input variable, the more useful the input variable is. According to this approach, to select useful variables, PCC between $V_{RMS-Radial}$ and variables acquired from flow, pressure and temperature sensors were computed. The results are presented in the Figure 5. Among the variables considered, outlet pressure, outlet flow, inlet and bleed pressure are good potential candidates for input variables.

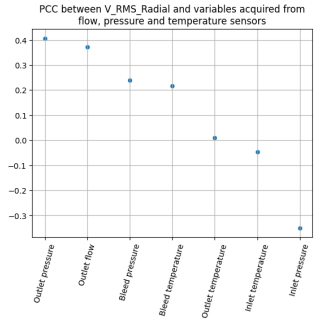


Figure 5. Pearson correlation coefficients.

One more factor that influences the choice of input variable is that the number of variables will be crucial in the context of training duration for models [35]. Generally, the more variables are chosen, the longer training lasts. This aspect harmonizes with the business requirement to minimize the amount and cost of sensors to be mounted.

Taking into consideration the calculated PCC values, the industrial implementation point of view and training time of models, it was decided to choose three sets of input variables:

- Outlet pressure;
- Outlet pressure, outlet flow;
- Outlet pressure, outlet flow, bleed pressure.

Other feature selection techniques such as linear regression with regularization L1 [36], kNN, DT, BME, XGBoost, RF [37] and RFE [33] are also widely used. However, they are beyond scope of this article.

Moreover, in the production environment for projects involving the Internet of Things approach, computation resources could be a constraint for fast inference, because of the need to perform predictions with models implemented on edge devices [38,39]. The feature selection process will help to speed up both the training and prediction process. In the next sections, further analysis involving the influence of the number of variables on model prediction quality will be performed.

2.3. Preparation to Training and Evaluation Process

The raw data acquired from the measuring experiments require some attention in the context of preprocessing. The first step that involve variables that could be informative in the context of modeling was performed in Section 2.2. Such a truncated dataset should be further processed and analyzed to take into consideration what approach will lead to results that are reliable and could be useful in production.

2.3.1. Evaluation Metrics

Three metrics were chosen based on analyses performed in previous publications [40,41]:

- Mean absolute value (MAE), which will provide a rough insight on raw prediction errors;
- Mean absolute percentage error (MAPE) which will provide an insight into results in the context of an evaluation of the results with respect to reference values;
- Mean squared error (MSE) that provides a good insight into outliers in prediction error sets.

A comparison of all the described metrics will provide an understanding of prediction errors that will lead to selecting the best model. Other metrics that are useful in the context of prediction models for time series like Log Mean Squared Error Ratio (LMR) and Median Relative Absolute Error (MdRAE) are also commonly used. However, they are beyond the scope of this article.

2.3.2. Dataset Preparation

The analysis performed in Section 2.2 leads to the conclusion that $V_{RMS-Radial}$ is selected as the input, and outlet pressure, outlet flow and bleed pressure are selected as the output variables. It was decided to feed the models with data taken from time windows of a fixed length w, to predict the next sample of $V_{RMS-Radial}$. Figure 6 presents this concept. A split was performed for the training and test dataset with a proportion of 80:20, respectively. The data preparation process does not involve the validation dataset. Only the training and test dataset were constructed. Such a decision was made due to the fact that the preparation of the dataset was quite complicated due to the multiplicity of the types of models used. For the analysis using dense neural networks in Section 2.4.1, the extracted time windows were randomly combined. While using recurrent neural networks (RNNs) and long-short-term memory neural networks (LSTMs) in Sections 2.5.2 and 2.5.3, there was no randomization of extracted time windows. In these cases, the border of the train-test split of the time windows was placed at a time of 80% from the beginning of the experiment (so 20% from the end of the experiment). Datasets for dense-, recurrent- and long short-term networks were standardized based on parameters obtained on their training sets. The analogical procedure was performed for each experiment separately. As data for all experiments was concatenated along the time axis, careful handling of the boundary conditions was performed, so that none of the extracted time windows overlapped over two experiments. Figure 7 schematically shows the dataset preparation for dense networks. Figure 8 presents it for recurrent and long short-term memory networks.

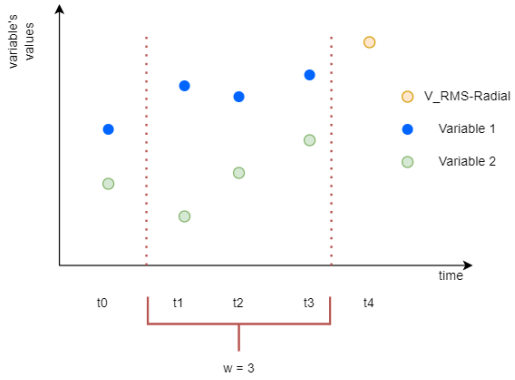


Figure 6. Illustration of aggregating samples from dataset into time window as input variables and picking the next sample of $V_{RMS-Radial}$ as the output variable of the model.

The last approach to prepare the dataset for learning models involves preparation for the training and evaluation of ensemble models, specifically stacking (blending) models. To ensure that blender (strong learner "blending" outputs of ensembles) is trained and evaluated in a way that will not make it positively biased, the training dataset must contain only test data for each of ensemble models. In turn, the test data set must also include only test data for ensemble models. As a result, for each measuring experiment data, only the last 20% of samples of time series were taken into consideration because this data is a test set for RNNs/LSTMs. From this 20%, only values that could be taken into consideration for training blender are examples that were not part of the training nor test set for dense networks. Such carefulness while constructing the dataset prevents data leakage from occurring. Trained blenders do not involve statefulness, therefore the resulting dataset could be randomly sampled in common way. Eighty (80)% of the available data are used for

training, and 20% of the available data are used for testing. Standardizing data undergoes the same method as previously outlined. Standardization parameters are estimated on the training dataset and applied to the whole dataset. An illustration of the construction of the dataset for blender training is presented in Figure 9.

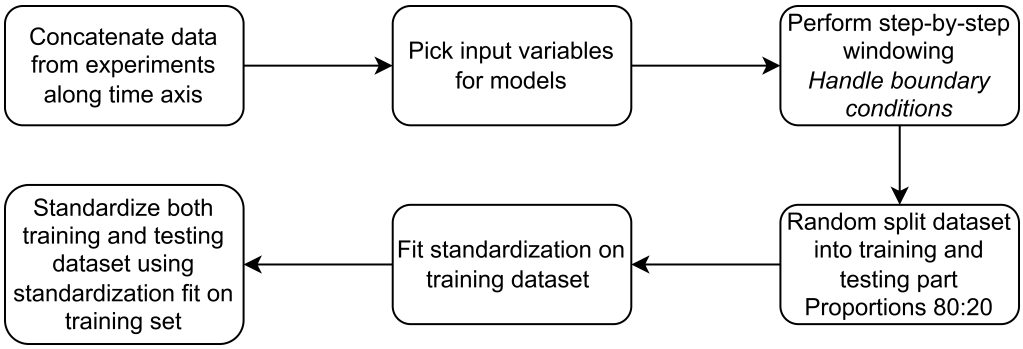


Figure 7. Scheme presenting dataset preparation for training and evaluation of dense models. Only data from pumps without faults is taken into consideration.

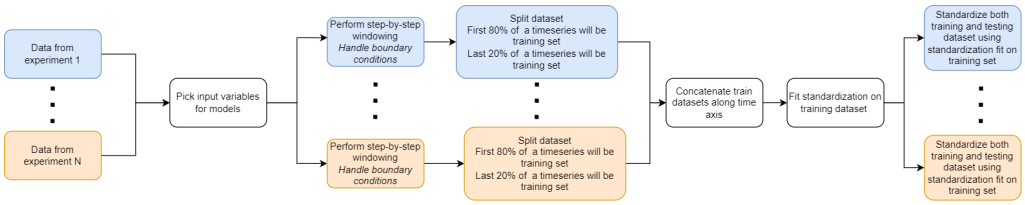


Figure 8. Scheme presenting dataset preparation for training and evaluation of recurrent models. Only data from pumps without faults is taken into consideration.

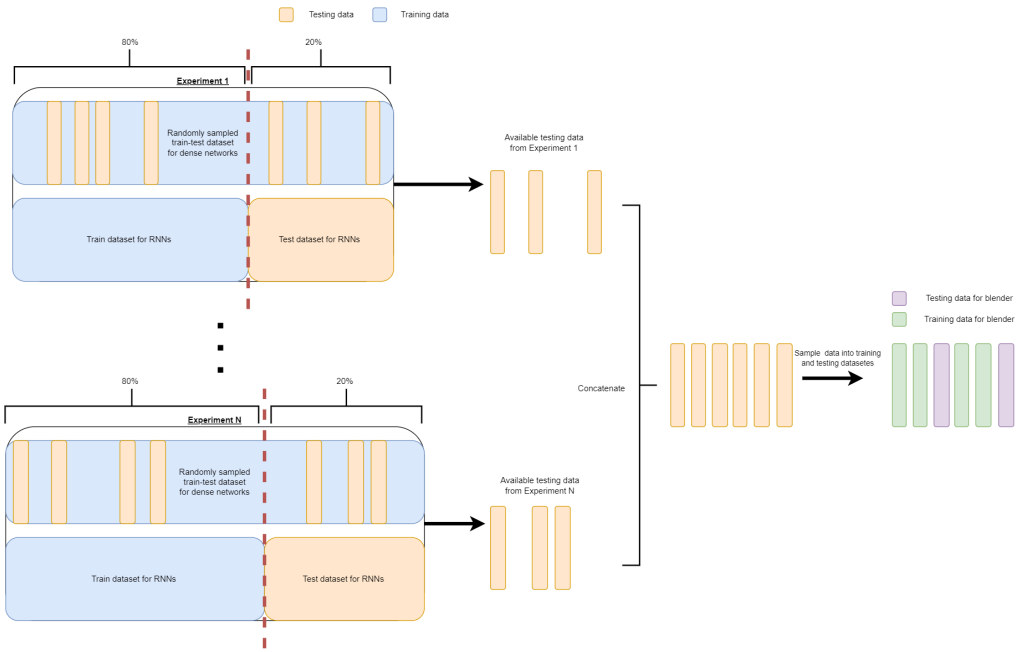


Figure 9. Process of preparation of the dataset for the purpose of training blender.

2.3.3. Neural Network Architectures and Training Process

For the purpose of the article, the analyzed neural network architectures were obtained by using grid search [42]. An alternative and even preferred method to grid search is random search [43]. However, for the purpose of the present article, grid search was chosen, to perform more extensive analysis on the impact of the hyperparameters of the network

on the final results. Table 3 presents a set of architecture hyperparameters to be explored. The number of input neurons of the network is determined by length of time window and number of input variables. For example, two variables with a time window of length 4 gives an input vector of size 8 ($2 \times 4 = 8$), so the input layer contains eight neurons. Regarding activation functions of the hidden neurons, defaults provided by the TensorFlow library were used. That is, for dense networks, ReLU [44] was used, and for RNNs and LSTMs hyperbolic tangent [45] was applied. For activation function of the output layer, ReLU was chosen, because the output of the network ($V_{RMS-Radial}$) is always non-negative.

Table 3. Parameters chosen for performing grid search while training neural networks.

Parameter	Values
Length of time window	4, 8, 16, 32
Number of hidden layers	1, 2, 3, 4, 5
Number of neurons in layer	5, 10
Variable sets	[Outlet pressure], [outlet pressure, outlet flow], [Outlet pressure, outlet flow, bleed pressure]
Total	96 combinations

Training neural networks performed in TensorFlow within the article was carried out using the setup presented in Table 4. Each choice has its rationale in a conceptual or implementation manner.

Table 4. Training setup.

Property	Value	Rationale
Optimizer	rmsprop [46]	TensorFlow default
Cost function	Huber loss	Marginalize errors coming from outliers [47]
Batch size	32	Good starting point for further research [48]
Epochs	500	Big value constrained to be regularization
Regularization	Early Stopping [43]	Efficient and simple to implement
Patience	20	Tweaked empirically

2.4. Regression

One of the approaches used for predicting future values in time series is regression [49]. In the described case, as an input, regression models take variables selected in Section 2.2 and give $V_{RMS-Radial}$ as an output. A modeling concept in regression manner involves selecting previous variables' values from a time-window of a fixed length.

$$V_{RMS-Radial}(t_{i+1}) = f(F1_{t_i}, \ldots, F1_{t_{i-w+1}}, \ldots, FN_{t_{i+1}}, \ldots, FN_{t_{i-w+1}}) \quad (1)$$

where:

- t_i is discrete time of current sample
- w is time-window of fixed length
- N is number of features taken into account
- F is value of input feature used in model
- $V_{RMS-Radial}(t_{i+1})$ is the value of next sample to predict

In such an approach, the model is stateless; it does not preserve information from previous predictions.

2.4.1. Dense Neural Networks

Models that are capable of performing predictions of good quality, often outperforming classic machine learning prediction algorithms, are dense neural networks [50]. This ability was the motivation to use such a type of model instead of classic neural networks, however, the latter are easier to analyze machine learning models.The considered network is a feedforward neural network with possible hyperparameters that were presented in Table 3.

2.5. Forecasting

Future values of variables could be estimated with another set of techniques. These techniques are within the scope of forecasting. In the described case, forecasting models calculate the output as a function of variables selected in Section 2.2 and hidden states of the models. As an output, they provide the $V_{RMS-Radial}$ value. The modeling concept in forecasting is similar to that presented in Section 2.4. This approach involves selecting previous variables' values from a time-window of a fixed length.

$$V_{RMS-Radial}(t_{i+1}) = f(F1_{t_i}, \ldots, F1_{t_{i-w+1}}, \ldots, FN_{t_i}, \ldots, FN_{t_{i-w+1}}, h_{t_i}, \ldots, h_{t_{i-w+1}}) \quad (2)$$

where:

- t_i is discrete time of current sample
- w is time-window of fixed length
- N is number of features taken into account
- F is value of input feature used in model
- $V_{RMS-Radial}(t_{i+1})$ is value of next sample to predict
- h is hidden state of the model

Unlike the models presented in the Section 2.4, the presented forecasting models preserve information acquired from previous predictions.

2.5.1. Neural Networks with Memory

RNNs and LSTMs [51] are types of neural networks that are capable of learning system dynamics by storing hidden states inside the model. These types of networks were chosen for the article here, also because of their capability to detect anomalies [52].

Similarly to the analysis performed in Section 2.4.1, several models were trained by performing a grid search according to the parameters' sets in Table 3. Training and validation of the models were performed by using data from experiments for pumps without faults. The architecture of the networks was carefully analyzed in the context of forecasting quality. The best models for both the RNN and LSTM approach were selected and then compared taking training, testing and evaluation models on properly formed datasets.

2.5.2. Recurrent Neural Networks

RNNs are one of the first introduced neural networks, and were a base for formulating further concepts of neural networks with memory [53]. These networks have many successful applications. On the other hand, it has been proven that they suffers from a lack of long-term memory [54]. However, this limitation is not necessarily an obstacle in the context of forecasting in some applications [55].

2.5.3. Long Short-Term Memory Networks

LSTM networks have been used with success in forecasting and anomaly detection applications [56]. In contrast to RNNs, they are able to have long-term memory [54], which in some applications could be an advantage for LSTM over RNN networks. An example of such an application is a chatbot [57].

2.6. Ensemble Learning

After training several types of models, techniques that mix them for improvement of performance could be applied. One such technique is ensemble learning [58]. Analysis performed within this section involves the usage of simple methods, such as computing response as mean or median of ensembles' responses, and more complex ones such as blending. Figure 10 schematically shows how the final output of the ensemble is calculated in case of applying a mean- median- or blender-ensemble. Another kind of ensemble method is gradient boosting which will also be used and described in Section 2.6.3. Only

the best model of each kind is taken into consideration for the final output calculation as a base estimator.

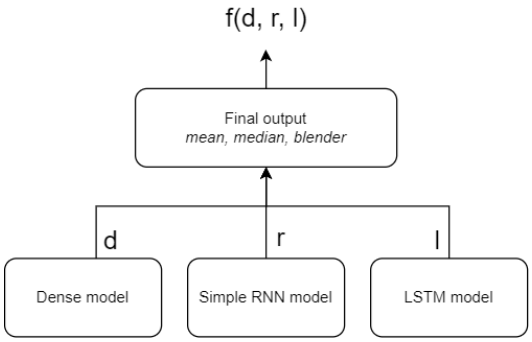


Figure 10. Way of calculating final output with use of the ensemble of models.

2.6.1. Simple Strategies

A simple method for determining the final output of model ensemble is to take an average or median of responses from the ensembles. A similar approach was proposed in article [59] for a task from the image processing domain.

2.6.2. Stacking

Instead of calculating output as a simple but intuitive mean or median, an approach that involves a more complex function could be applied. Such a function could be constructed by training a separate model that learns the function itself. This kind of model is called a blender [60] and an approach that involves this way of handling ensemble outputs is called stacking [61]. In the considered approach, the blender is a linear regression model [62].

2.6.3. Gradient Boosting

Instead of mixing outputs of a couple of models into a single prediction, another model could be trained to correct the errors of the original model. Such a way of correcting prediction errors with a model dedicated to this task is called gradient boosting (GB) [63]. This technique has several practical applications. One of the examples is presented in an article where the authors used GB for characterizing a shallow marine reservoir [64]. Another example is shown in anti-money laundering practices in cryptocurrencies [65].

For the purpose of this article, the best model of each presented kind (dense, RNN, LSTM) was taken as a base predictor, and a linear regression model will act as an error predictor. The final output will be the sum of predictions of the base model and error predictor.

3. Results and Discussion

The presentation of the results and discussion is focused on three aspects:

1. Complexity—in terms of model size and implementation
2. Performance—presentation of model quality metrics
3. Training process—comparison and possibilities of improvement

Taking all above factors into consideration enables us to indicate the strengths of the different presented approaches and compare them in an orderly manner.

3.1. Complexity

Analysis of the complexity should be carried out using different criteria for neural networks and ensembles of the models. In this section, the value of the MAPE metric evaluated on the test set will be used while comparing models. Section 3.2 includes a more comprehensive discussion of other performance metrics.

3.1.1. Neural Networks

For neural networks, the most crucial aspect is the size of the network. The more neurons the network has, the more complex it becomes. The number of neurons is strictly related to inference time and the use of computational resources at the place where the neural network is located [66]. Based on the parameter sets presented in Table 3, grid search was performed and all available network architectures were implemented and trained. A comparison of the architectures in confrontation with MAPE metrics is presented in Figures 11–13. MAPE metrics were chosen because, at first glance among other used metrics, they give the most insight into results, without knowing the predicted values. A detailed discussion of the metrics used is presented in Section 2.3.1.

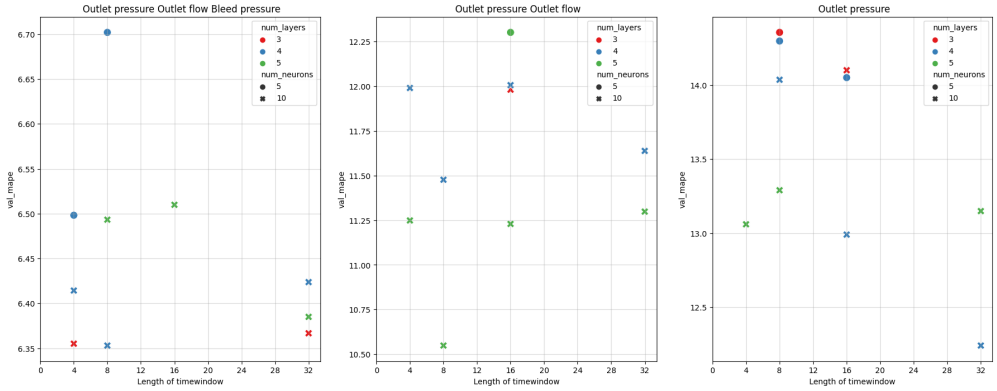


Figure 11. Quality metrics comparison for top 10 dense models for each selected set of model input variables.

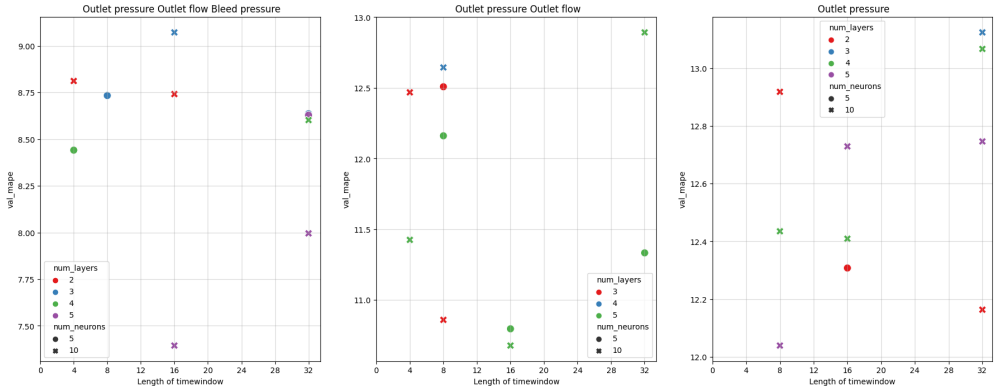


Figure 12. Quality metrics comparison for top 10 RNN models for each selected set of model input variables with respect to network architecture.

Several conclusions could be drawn based on the plots presented in Figures 11–13. For all types of network, the most abundant set of the input variables generally results in significantly better performance. Also, for all types of models and for all sets of input variables, the best model is characterized by 10 neurons in the hidden layers. Furthermore, for architecture of this property, in almost all cases, the value of MAPE of the best models is significantly lower than for networks with five neurons in the hidden layers. The exceptions are RNN with the [output pressure, outlet flow] input variables set, and LSTM with the [output pressure, outlet flow, bleed pressure] input variables set. The length of time window which impacts complexity in context of the number of input neurons is very varied for the best models. In Figure 11, the top three models do not differ much in terms of performance, but are characterized by three different numbers of input neurons. For almost the best model of each type and each set of input variables, the best one is the most complex. That is, it has 10 neurons in each hidden layer and five hidden layers. The only exceptions are the dense model with the [output pressure, outlet flow, bleed pressure] variable set and

the LSTM model with the [output pressure, outlet flow] variable set, which both have four hidden layers.

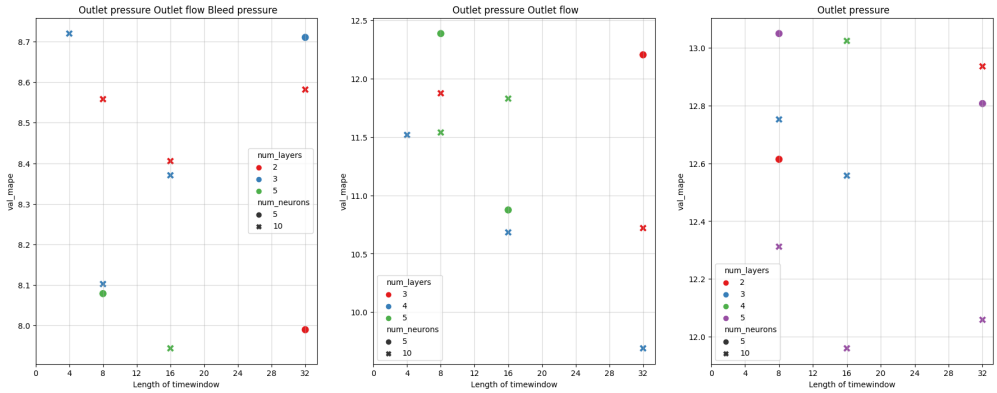


Figure 13. Quality metrics comparison for top 10 LSTM models for each selected set of model input variables with respect to network architecture.

The degree of complexity of the implementation is comparable for all presented models. The implementation with Python using TensorFlow enables us to change the network type and its hyperparameters in a very convenient, fast and flexible way.

Taking the above analysis into consideration, the best neural network model is characterized by the properties presented in Table 5. This model has slightly worse performance than the best model among all available, but it has significantly simpler architecture.

Table 5. Parameters of the best model in terms of complexity.

Property	Value
Number of hidden layers	3
Number of neurons in hidden layers	10
Length of time window	4
Variable set	[Output pressure, outlet flow, bleed pressure]

3.1.2. Ensemble Learning

The complexity of the ensembles of the models should be assessed not only in terms of complexity of the underlying models, but also in terms of complexity that is implied by implementing the ensemble itself. Furthermore, the ensembles can only be compared with each other because, as described in Section 2.3.2, ensembles and neural networks use separate pieces of a dataset to train and evaluate them.

Using the stacking strategy implies the need for training a separate model that will mix (blend) outputs of the ensembles into single one. The outputs of the best dense, RNN and LSTM models in terms of test MAPE metrics presented in Tables 6–8 were taken into consideration. In the analyzed case, linear regression was taken as a blender model. Therefore, it allows for the assessment of which output of which model impacts the final output the most. The higher the coefficient of the linear model, the more impact it has on the final result. The trained model coefficients sorted in descending absolute values are as follows:

- Dense model: 1.1849
- RNN: −0.2081
- LSTM: −0.0610

Table 6. Comparison of metrics evaluated during training for three best dense models for each of input variables set.

Num Epochs	MAPE	MAE	Test MAPE	Test MAE	Loss	Num Layers	Num Neurons	Num of Input Timepoints	Input Cols
358	6.8548	0.1077	6.353	0.0752	0.0363	4	10	8	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
453	6.9464	0.1078	6.3551	0.0766	0.0356	3	10	4	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
339	6.841	0.0859	6.3667	0.0778	0.014	3	10	32	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
500	10.672	0.147	10.5487	0.1185	0.0413	5	10	8	['Outlet pressure', 'Outlet flow']
500	11.5539	0.1558	11.2289	0.1301	0.0418	5	10	16	['Outlet pressure', 'Outlet flow']
500	11.4723	0.1536	11.2484	0.1272	0.0416	5	10	4	['Outlet pressure', 'Outlet flow']
500	12.4225	0.1452	12.2398	0.14	0.0232	4	10	32	['Outlet pressure']
441	13.3095	0.1745	12.9897	0.147	0.0463	4	10	16	['Outlet pressure']
337	13.3476	0.1751	13.0594	0.147	0.0459	5	10	4	['Outlet pressure']

Table 7. Comparison of metrics evaluated during training for three best RNN models for each of input variables set.

Num Epochs	MAPE	MAE	Test MAPE	Test MAE	Loss	Num Layers	Num Neurons	Num of Input Timepoints	Input Cols
40	7.5436	0.1042	7.3943	0.1071	0.0327	5	10	16	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
50	6.782	0.0802	7.9953	0.113	0.0106	5	10	32	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
34	7.198	0.1044	8.441	0.1197	0.0326	4	5	4	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
42	14.7205	0.1792	10.6782	0.1501	0.0435	5	10	16	['Outlet pressure', 'Outlet flow']
56	14.6767	0.179	10.7966	0.1451	0.0433	5	5	16	['Outlet pressure', 'Outlet flow']
105	13.892	0.1695	10.8598	0.1486	0.0417	3	10	8	['Outlet pressure', 'Outlet flow']
23	14.8307	0.1843	12.0391	0.1674	0.0448	5	10	8	['Outlet pressure']
27	15.612	0.1699	12.1633	0.1685	0.0248	2	10	32	['Outlet pressure']
22	15.9429	0.1915	12.3078	0.1718	0.0465	2	5	16	['Outlet pressure']

Table 8. Comparison of metrics evaluated during training for three best LSTM models for each of input variables set.

Num Epochs	MAPE	MAE	Test MAPE	Test MAE	Loss	Num Layers	Num Neurons	Num of Input Timepoints	Input Cols
58	6.9488	0.1033	7.9439	0.1117	0.0326	5	10	16	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
71	7.4563	0.0858	7.9897	0.1169	0.0114	2	5	32	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
112	7.4918	0.1035	8.0788	0.1183	0.0329	5	5	8	['Outlet pressure', 'Outlet flow', 'Bleed pressure']
104	13.8395	0.1548	9.6899	0.1344	0.0212	4	10	32	['Outlet pressure', 'Outlet flow']
74	13.9802	0.1774	10.6833	0.1476	0.0431	4	10	16	['Outlet pressure', 'Outlet flow']
56	14.0297	0.1566	10.7201	0.1417	0.0218	3	10	32	['Outlet pressure', 'Outlet flow']
40	15.5522	0.1916	11.9598	0.1707	0.0464	5	10	16	['Outlet pressure']
32	14.1244	0.1573	12.0584	0.1674	0.0222	5	10	32	['Outlet pressure']
25	15.777	0.1901	12.3115	0.1681	0.0463	5	10	8	['Outlet pressure']

The conclusion for stacking ensemble is that the dense model has the greatest impact on the final value of the ensemble. This conclusion is coherent with the fact that the dense model was the best one in terms of MAPE metrics amongst all of the neural network models.

Computing the output of the blender requires the computation of the outputs of models that are part of the ensemble. As there are three models (dense, RNN, LSTM), using the stacking ensemble strategy heavily impacts the computational resources and inference time.

The last used ensemble involves GB. Within this approach, we selected one model of each type (dense, RNN, LSTM) and trained a separate linear regression model as a corrector of prediction error of the base model. Three GB ensembles were created, where each of them consists of one base neural network model followed by a linear regression model. The GB approach is less complex than simple strategies in terms of the number of models involved. To calculate the output of the simple strategies ensemble, the outputs of three models (dense, RNN, LSTM) must be computed, while in the GB approach two models must be used: one neural network model and one linear regression model.

Among all presented ensembles, in terms of MAPE metrics selected among experiments for non-damaged pumps, which are presented in Table 9, the best one is the median ensemble. It outperforms other ensembles in all experiments except the reference tests with check valve, where GB Dense performed slightly better. In addition, simple strategies, stacking and GB Dense perform comparably well on all experiments, while GB RNN and GB LSTM perform significantly worse.

Table 9. Metrics for simple strategies and gradient boosting ensembles for each evaluated on both training and test data for ensembles.

Metric	GB Dense	GB RNN	GB LSTM	Median	Mean	Linear Regression
Experiment: Reference tests v1						
MAE	0.0711	0.0918	0.0983	0.067	0.0685	0.0764
MAPE	6.02	7.89	8.67	5.66	5.83	6.61
MSE	0.0101	0.0149	0.018	0.0094	0.0095	0.0101
Experiment: Reassembling v1						
MAE	0.0773	0.1048	0.0997	0.0754	0.0764	0.0771
MAPE	5.27	7.03	6.71	5.05	5.12	5.26
MSE	0.0122	0.0227	0.0203	0.0127	0.013	0.0122
Experiment: Reference tests v2						
MAE	0.086	0.1193	0.1021	0.0825	0.0833	0.0824
MAPE	6.98	9.94	8.56	6.64	6.71	6.82
MSE	0.0158	0.0245	0.0199	0.0156	0.0159	0.0134
Experiment: Reference tests with check valve						
MAE	0.0957	0.114	0.1393	0.0982	0.098	0.1002
MAPE	8.6	10.2	13.29	8.89	8.94	9.25
MSE	0.017	0.0238	0.0271	0.017	0.0165	0.0163
Experiment: Axial play fault						
MAE	0.6196	0.799	0.9081	0.7705	0.75	0.5925
MAPE	28.65	37.06	42.21	35.79	34.78	27.29
MSE	0.4165	0.6816	0.8713	0.6275	0.5984	0.3873
Experiment: Timing plate fault						
MAE	0.3519	0.3407	0.3406	0.3131	0.3258	0.3646
MAPE	19.79	19.11	19.12	17.51	18.25	20.52
MSE	0.1493	0.1407	0.1418	0.1198	0.1287	0.1575

Table 9. *Cont.*

Metric	GB Dense	GB RNN	GB LSTM	Median	Mean	Linear Regression
Experiment: Swash plate fault						
MAE	2.2711	2.3352	2.1457	2.2727	2.2522	2.245
MAPE	65.0	66.83	61.37	65.04	64.45	64.25
MSE	5.1945	5.4953	4.6474	5.2011	5.1085	5.0758
Experiment: Bearing fault						
MAE	0.1397	0.1943	0.1555	0.1726	0.1573	0.1192
MAPE	10.8	14.67	11.57	13.03	11.88	9.2
MSE	0.0304	0.053	0.0399	0.0407	0.0355	0.0226

In terms of implementation complexity, the easiest ones to implement are simple strategies: mean- and median-ensemble. The only thing to do in that case is to take outputs of the ensemble models and compute mean or median from them, depending on the type of ensemble used. This approach does not involve training any additional model. More complex strategies, such as stacking and GB, require training one additional model per ensemble. Additional effort is required in both cases to construct separate training and test datasets. Therefore, for stacking and GB, the implementation complexity is comparable.

As a conclusion, taking complexity of implementation, impact on computational resources and MAPE metric value, the GB dense ensemble is the best choice. It does not have the best MAPE value, nor the simplest implementation, but it has the lowest impact on computational resources. While considering such a compromise, that choice is justified.

3.2. Performance

Relying on single metrics is useful for simpler analysis of the models. However, taking more metrics into consideration could give more insight into performance of the trained models. The analysis of the performance of neural networks and ensemble models is different. As stated in Section 2.3.2, they are using different pieces of the dataset for training and evaluation, so their performance cannot be compared to each other.

3.2.1. Neural Networks

To check the legitimacy of the trained models in context of assessing condition of the pump, metrics were computed on the whole experiments with both damaged and not damaged devices. A distinction between four types of damage was indicated as mentioned in Section 2.1. The models were trained on data from experiments where the pump had no damage, so it is expected that they would obtain a significant difference between the prediction quality calculated on the scenarios for the pump with and without damage. The best models for the purpose of the following analysis were chosen by the lowest test MAPE value among Tables 6–8 for dense, RNN and LSTM models, respectively. The metrics computed for these models on all experiments are presented in Tables 10–12. These results are the basis for drawing several conclusions.

The more input variables the model has, the greater the difference between metrics for scenarios for the pump with and without damage. For many models that take one or two variables on the input, it emerges that the bearing fault scenario has better fitness metrics than scenarios that involve pump without faults.

A high MSE value in combination with low MAPE and MAE values means for all types of models for experiment called Reference tests v1, that the outliers are present in the experiment. However, there are some models, for example the best LSTM model with one input variable, that are able to handle outliers better than models that have lower values of MAE and MAPE metrics. A similar situation appears for an experiment called Bearing fault. However, in that case, the MSE value is high for all types of models for all number of input variables.

Table 10. MAPE, MAE and MSE metrics for whole dataset evaluated for best dense model for following sets of input variables: 1 var—[Outlet pressure], 2 var—[Outlet pressure, Outlet flow], 3 var—[Outlet pressure, Outlet flow, Bleed pressure].

Experiment	$MAPE_{1var}$	MAE_{1var}	MSE_{1var}	$MAPE_{2var}$	MAE_{2var}	MSE_{2var}	$MAPE_{3var}$	MAE_{3var}	MSE_{3var}
Reference tests v1	9.85	0.12	0.47599	13.27	0.23	15.3615	5.46	0.15	15.4813
Reassembling v1	12.52	0.19	0.10032	9.45	0.14	0.11833	5.37	0.09	0.01366
Reference tests v2	11.16	0.13	0.09743	10.09	0.12	0.07476	7.27	0.08	0.03478
Reference tests with check valve	18.42	0.17	0.17454	13.23	0.13	0.11489	7.24	0.08	0.11464
Axial play fault	35.63	0.7	0.54233	35.09	0.67	0.57356	22.11	0.45	0.29764
Timing plate fault	27.9	0.46	0.25225	34.78	0.56	0.39376	19.98	0.33	0.14984
Swash plate fault	62.3	1.85	3.57135	59.99	1.79	3.31754	57.75	1.72	3.14876
Bearing fault	8.66	0.23	39.9357	7.6	0.21	39.0417	14.49	0.29	39.3965

Table 11. MAPE, MAE and MSE metrics for whole dataset evaluated for best RNN model for following sets of input variables: 1 var—[Outlet pressure], 2 var—[Outlet pressure, Outlet flow], 3 var—[Outlet pressure, Outlet flow, Bleed pressure].

Experiment	$MAPE_{1var}$	MAE_{1var}	MSE_{1var}	$MAPE_{2var}$	MAE_{2var}	MSE_{2var}	$MAPE_{3var}$	MAE_{3var}	MSE_{3var}
Reference tests v1	8.18	0.18	15.7687	11.46	0.21	15.3085	6.27	0.16	15.8153
Reassembling v1	20.79	0.29	0.13925	13.54	0.19	0.12856	6.03	0.09	0.02502
Reference tests v2	17.53	0.19	0.07747	16.72	0.18	0.07895	9.53	0.09	0.04795
Reference tests with check valve	14.27	0.13	0.10543	14.49	0.14	0.13944	8.05	0.09	0.13261
Axial play fault	42.44	0.82	0.76792	37.82	0.72	0.52341	42.02	0.77	0.68488
Timing plate fault	34.58	0.57	0.31191	34.29	0.56	0.30798	15.8	0.26	0.04874
Swash plate fault	64.04	1.9	3.76273	62.72	1.87	3.62271	59.32	1.77	3.27422
Bearing fault	9.04	0.23	39.9641	8.17	0.22	39.8108	13.9	0.28	39.1801

Table 12. MAPE, MAE and MSE metrics for whole dataset evaluated for best LSTM model for following sets of input variables: 1 var—[Outlet pressure], 2 var—[Outlet pressure, Outlet flow], 3 var—[Outlet pressure, Outlet flow, Bleed pressure].

Experiment	$MAPE_{1var}$	MAE_{1var}	MSE_{1var}	$MAPE_{2var}$	MAE_{2var}	MSE_{2var}	$MAPE_{3var}$	MAE_{3var}	MSE_{3var}
Reference tests v1	7.76	0.1	0.48272	13.34	0.15	0.49814	7.92	0.18	15.1508
Reassembling v1	20.47	0.29	0.16713	12.32	0.18	0.16963	6.1	0.09	0.09628
Reference tests v2	15.66	0.19	0.03894	12.15	0.14	0.05299	13.54	0.17	0.09184
Reference tests with check valve	14.12	0.13	0.11306	18.57	0.18	0.10934	17.74	0.17	0.17386
Axial play fault	42.14	0.81	0.71375	37.15	0.71	0.51511	39.53	0.74	0.67366
Timing plate fault	34.32	0.56	0.34247	30.87	0.5	0.28487	15.72	0.26	0.06824
Swash plate fault	64.04	1.91	3.78185	61.84	1.84	3.58137	59.63	1.78	3.30235
Bearing fault	9.07	0.23	39.3846	9.66	0.24	39.4206	11.54	0.26	39.0876

For any of the best-considered models for all experiments except Swash plate fault, MAE cannot be a metric that could act as the single one to be used to determine the condition of the monitored pump. MAE units are the same as the vibration velocity, that is $\frac{\text{mm}}{\text{s}}$. Even under the assumption that the error is always positive, the value of MAE is not significant enough to indicate according to ISO norm [3] that the condition of the machine has changed. The situation looks different in the case of the Swash plate fault experiment, where MAE is higher than 1.7 $\frac{\text{mm}}{\text{s}}$ for every model. This value is significant. Computing MAE between readings from the reference sensor and predicted model output could be a good indicator that the pump has swash plate damage. The MAE value is significant enough to both indicate the change in machine condition according to the ISO norm and even precisely point out that the swash plate is damaged.

The best model that could be chosen considering the metrics presented in Tables 10–12 is the dense network with three input variables. There are a couple of other models that have larger error metrics for the Axial play fault scenario—even models with fewer input variables or models of different type. However, on average, a dense network with three input variables emphasizes the most differences in error metrics for scenarios with damaged and non-damaged pumps.

The values of the discussed metrics are reflected in the plots shown in the Figures 14–16. In addition to the results discussed so far, in the mentioned Figures for experiment called Reference tests v2, it is visible that all presented models were capable of capturing the signal dynamics. For the part of the experiment where there was a step down, the models followed that step down.

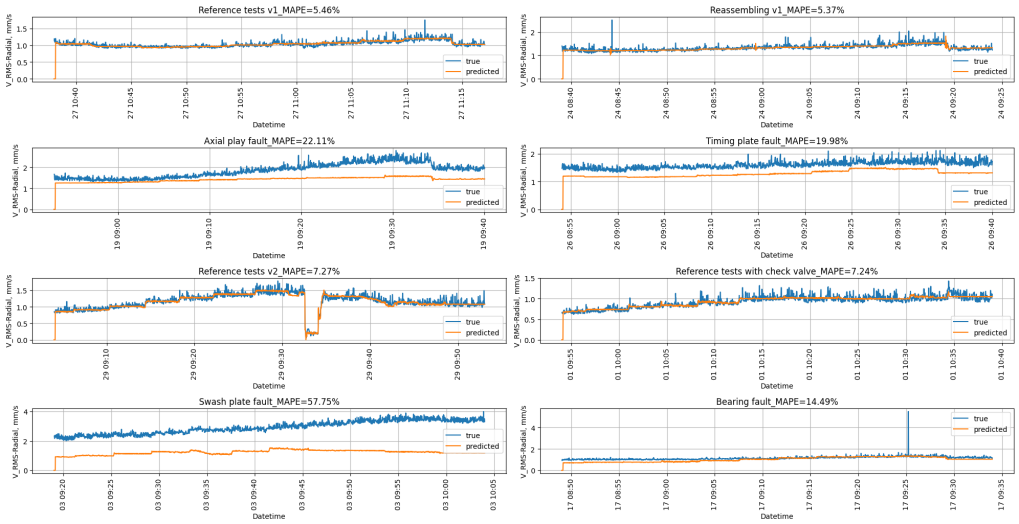


Figure 14. $V_{RMS-Radial}$ generated by best dense model for all experiments. For clarity, median filter (3 samples) was applied.

3.2.2. Ensemble Learning

For ensembles of models, metrics were calculated on data from both the training and test dataset for experiments with pump without damage. In terms of MAPE value, the median ensemble performs best among others presented in Table 9. Generally, all of the ensembles performed comparably well, except GB RNN and GB LSTM, which are characterized by significantly worse metric values compared to the other ensembles. The value of a MSE metrics is on a comparable level or even lower than the MAE value. This could lead to the conclusion that outliers are not present in the evaluated dataset. The MAE values for experiments with damaged pumps are several times greater than the MAE metrics for experiments with non-damaged pumps. This property applies to all ensembles. Similarly to the discussion carried out in Section 3.2.1, only for the Swash plate fault experiment, MAE metrics can be a single factor that could be relied on when assessing whether the pump is damaged or not. The MAE value for that experiment is higher than

2.1 $\frac{\text{mm}}{\text{s}}$, which is significantly higher than for other experiments. It allows one to assess the change in machine condition in confrontation with the ISO norm. In addition, the MAE value for Swash plate fault calculated using any of the implemented ensembles could be a single metric that could indicate the swash plate damage.

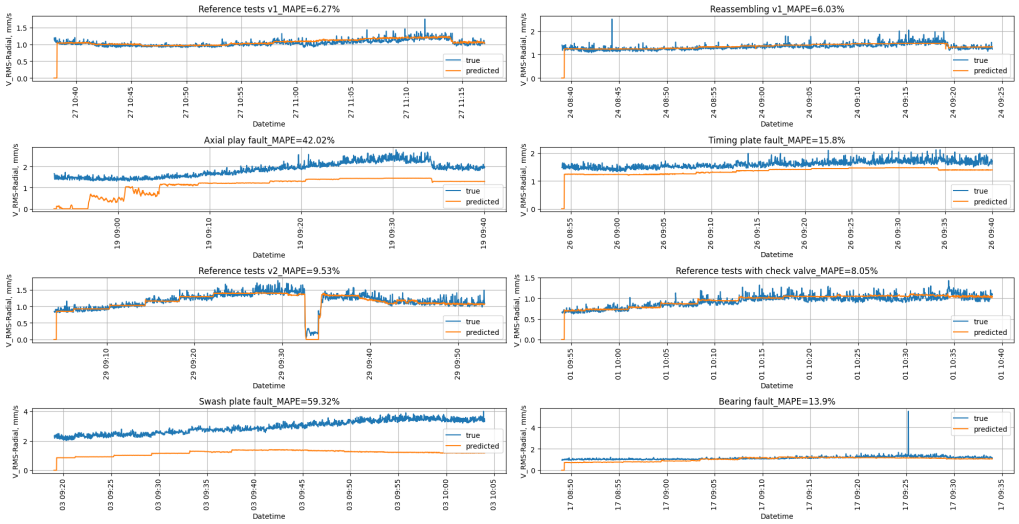


Figure 15. $V_{RMS-Radial}$ generated by best RNN model for all experiments. For clarity, median filter (3 samples) was applied.

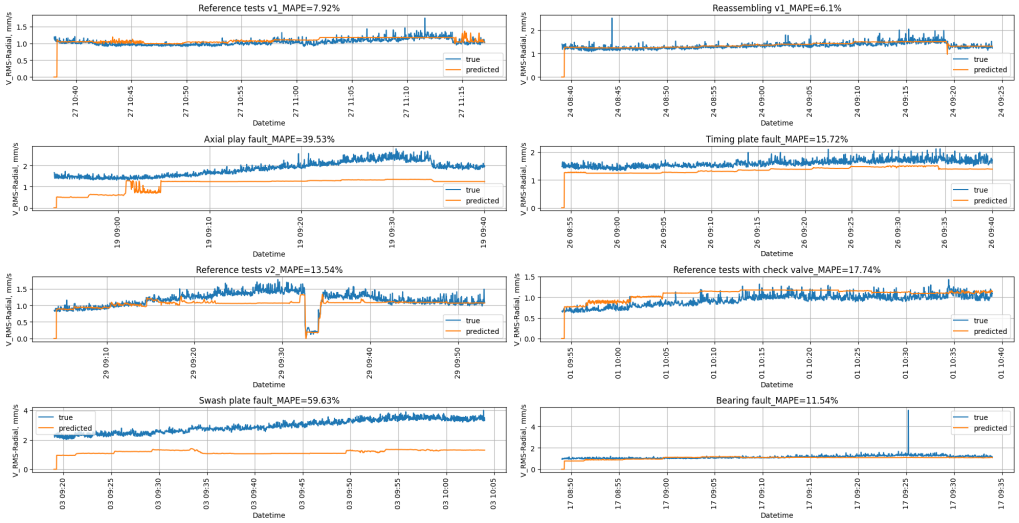


Figure 16. $V_{RMS-Radial}$ generated by best LSTM model for all experiments. For clarity, median filter (3 samples) was applied.

3.3. Training Process

All of the considered neural network models follow the same training setup presented in Table 4. While developing ensemble models, the only training process was the training of linear regression models. That training follows a different setup and scheme than training neural network models. The procedure for training linear regression models used in the article is presented in the scikit-learn documentation [67].

Tables 6–8 show that dense networks are significantly characterized by the highest number of training epochs. Some of the dense models even reached the maximum number of training epochs. For such models, training could be continued because there is a probability that performance of the network will increase after a longer training time. Training RNN networks requires fewer epochs than training LSTM networks on average. There is no general rule that the larger the model, the longer the training lasts, however.

The last important aspect in the context of training networks is training stability. Figure 17 presents the training process for the best models of each type analyzed. The best model here means the model with the lowest MAPE value computed on test set. The properties of the best models of each type are shown in Tables 6–8. The least stable training process took place for the LSTM model. The validation loss in this case is noisy and it is hard to conclude the shape of the downtrend of the loss curve. In addition, in Figure 17, the loss calculated over the test dataset for the dense and RNN model is lower than the training loss. This phenomena can occur due to the specifics of the dataset, especially when the test dataset contains more "easy" examples, or regularization is applied as presented in the article [68].



Figure 17. Training process for the best model of each type.

4. Conclusions and Further Research

The article presents different approaches for the vibration velocity prediction task for an axial piston pump.

Among the two available vibration sensors, the most informative was selected as the reference signal that is threatened as an output of the trained models. Furthermore, according to the expert knowledge and requirements of possible system implementation in industry, three sets of variables measured during the experiments carried out were analyzed and taken into consideration as input to the models. Datasets for training and evaluation of models of each kind and all approaches presented were carefully designed, so that no information leakage will occur.

The article includes a comprehensive variety of deep learning modeling approaches. Specifically, dense, RNN and LSTM networks were used. Moreover, a fusion of these methods was applied. The methods are simple strategies, such as mean and median ensemble, stacking and gradient boosting, as more complex strategies to implement ensemble of models.

Our study reveals that by using the MAPE metric it is possible to determine whether a pump is damaged or not. In addition, among all the implemented approaches, the use of the best neural network model of any type with any considered set of the input variables allows to point out swash plate damage using MAE metric computed between reference sensor indications and model outputs. The same property applies to all implemented ensemble strategies.

The developed models are capable of prediction for only one sample in advance. However, it is a good starting point for developing models that can predict more steps forward. Therefore, they will be well-suited for predictive maintenance applications. The learning procedure of the analyzed models could be tweaked. More learning rates, learning rate

scheduling, different batch sizes, different optimizers, higher patience parameter, or more training epochs could be taken into consideration. On the other hand, because of the many degrees of freedom, a study on the effects of tweaking learning parameters on models performance is not trivial. Different network types and architectures could also be a subject of further research. Transfer learning for a type of load other than constant with step changes, for example sine, could be carried out.

Author Contributions: Conceptualization, methodology, software, P.F.; writing—original draft preparation, P.F.; writing—review and editing, A.C.; Conceptualization, P.R. All authors have read and agreed to the published version of the manuscript.

Funding: The research is co-financed under the Polish Ministry of Science and Higher Education Program "Doktorat wdrożeniowy".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Restrictions apply to the availability of these data. Data was obtained from PONAR Wadowice S.A. and are available from PONAR Wadowice S.A. with the permission of PONAR Wadowice S.A.

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

References

1. Ran, Y.; Zhou, X.; Lin, P.; Wen, Y.; Deng, R. A Survey of Predictive Maintenance: Systems, Purposes and Approaches. *arXiv* **2019**, arXiv:1912.07383. https://doi.org/10.48550/arXiv.1912.07383.
2. Minescu, M.; Marius, S.; Avram, L. Fault detection and analysis at pumping units by vibration interpreting encountered in extraction of oil. *J. Balk. Tribol. Assoc.* **2015**, *21*, 711–723.
3. *BS ISO 20816-8:2018*; Mechanical Vibration—Measurement and Evaluation of Machine Vibration. ISO: Geneva, Switzerland, 2022.
4. Kim, W.; Braun, J.E. Development, implementation, and evaluation of a fault detection and diagnostics system based on integrated virtual sensors and fault impact models. *Energy Build.* **2020**, *228*, 110368. [CrossRef]
5. Hulkkonen, M.; Veijola, T.; Kallio, A.; Andersson, M.; Valtonen, M. Measurement-Based Equivalent Circuit Model for Ferrite Beads. In Proceedings of the 2009 European Conference on Circuit Theory and Design, Antalya, Turkey, 23–27 August 2009; pp. 363–366. [CrossRef]
6. Cui, S.; Yin, Y.; Wang, D.; Li, Z.; Wang, Y. A stacking-based ensemble learning method for earthquake casualty prediction. *Appl. Soft Comput.* **2021**, *101*, 107038. [CrossRef]
7. Chen, C.H.; Tanaka, K.; Kotera, M.; Funatsu, K. Comparison and improvement of the predictability and interpretability with ensemble learning models in QSPR applications. *J. Cheminform.* **2020**, *12*, 19. [CrossRef] [PubMed]
8. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Available online: https://tensorflow.org (accessed on 12 Septemper 2023).
9. Chollet, F. Keras. 2015. Available online: https://github.com/fchollet/keras (accessed on 12 Septemper 2023).
10. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
11. Harris, C.R.; Millman, K.J.; van der Walt, S.J.; Gommers, R.; Virtanen, P.; Cournapeau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N.J.; et al. Array programming with NumPy. *Nature* **2020**, *585*, 357–362. [CrossRef]
12. McKinney, W. Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference, Austin, TX, USA, 28 June–3 July 2010; pp. 51–56. [CrossRef]
13. Hunter, J.D. Matplotlib: A 2D graphics environment. *Comput. Sci. Eng.* **2007**, *9*, 90–95. [CrossRef]
14. Waskom, M.L. Seaborn: statistical data visualization. *J. Open Source Softw.* **2021**, *6*, 3021. [CrossRef]
15. Vrignat, P.; Kratz, F.; Avila, M. Sustainable manufacturing, maintenance policies, prognostics and health management: A literature review. *Reliab. Eng. Syst. Saf.* **2022**, *218*, 108140. [CrossRef]
16. Achouch, M.; Dimitrova, M.; Ziane, K.; Sattarpanah Karganroudi, S.; Dhouib, R.; Ibrahim, H.; Adda, M. On Predictive Maintenance in Industry 4.0: Overview, Models, and Challenges. *Appl. Sci.* **2022**, *12*, 8081. [CrossRef]
17. Dreyfus, P.A.; Pélissier, A.; Psarommatis, F.; Kiritsis, D. Data-based model maintenance in the era of industry 4.0: A methodology. *J. Manuf. Syst.* **2022**, *63*, 304–316. [CrossRef]
18. Lasithan, L.G.; Shouri, P.V.; Rajesh, V.G. Defining Vibration Limits for Given Improvements in System Availability. In Proceedings of the Advances in Data-Driven Computing and Intelligent Systems, Gujarat, India, 21–23 September 2023; pp. 589–602.
19. Cao, S.; Zhang, Y.; Tian, H.; Ma, R.; Chang, W.; Chen, A. Drive comfort and safety evaluation for vortex-induced vibration of a suspension bridge based on monitoring data. *J. Wind. Eng. Ind. Aerodyn.* **2020**, *204*, 104266. [CrossRef]

20. Iqbal, M.Y.; Wang, T.; Li, G.; Li, S.; Hu, G.; Yang, T.; Gu, F.; Al-Nehari, M. Development and Validation of a Vibration-Based Virtual Sensor for Real-Time Monitoring NOx Emissions of a Diesel Engine. *Machines* **2022**, *10*, 594. [CrossRef]
21. Zhu, Y.; Zhou, T.; Tang, S.; Yuan, S. Failure Analysis and Intelligent Identification of Critical Friction Pairs of an Axial Piston Pump. *J. Mar. Sci. Eng.* **2023**, *11*, 616. [CrossRef]
22. Leite Coelho da Silva, F.; da Costa, K.; Canas Rodrigues, P.; Salas, R.; López-Gonzales, J.L. Statistical and Artificial Neural Networks Models for Electricity Consumption Forecasting in the Brazilian Industrial Sector. *Energies* **2022**, *15*, 588. [CrossRef]
23. Zonta, T.; da Costa, C.A.; Zeiser, F.A.; de Oliveira Ramos, G.; Kunst, R.; da Rosa Righi, R. A predictive maintenance model for optimizing production schedule using deep neural networks. *J. Manuf. Syst.* **2022**, *62*, 450–462. [CrossRef]
24. Guo, R.; Zhao, Z.; Huo, S.; Jin, Z.; Zhao, J.; Gao, D. Research on State Recognition and Failure Prediction of Axial Piston Pump Based on Performance Degradation Data. *Processes* **2020**, *8*, 609. [CrossRef]
25. Ying, P.; Tang, H.; Chen, L.; Ren, Y.; Kumar, A. Dynamic modeling and vibration characteristics of multibody system in axial piston pump. *Alex. Eng. J.* **2023**, *62*, 523–540. [CrossRef]
26. Ying, P.; Tang, H.; Ye, S.; Ren, Y.; Xiang, J.; Kumar, A. Dynamic modelling of swashplate with local defects in axial piston pump and coupled vibration analysis. *Mech. Syst. Signal Process.* **2023**, *189*, 110081. [CrossRef]
27. Lagare, R.B.; da Conceicao, M.A.; Rosario, A.C.A.; Young, K.L.; Huang, Y.S.; Sheriff, M.Z.; Clementson, C.; Mort, P.; Nagy, Z.; Reklaitis, G.V. Development of a Virtual Sensor for Real-Time Prediction of Granule Flow Properties. In *Computer Aided Chemical Engineering, Proceedings of the 32nd European Symposium on Computer Aided Process Engineering, Toulouse, France, 12–15 June 2022*; Montastruc, L., Negny, S., Eds.; Elsevier: Amsterdam, The Netherlands, 2022; Volume 51, pp. 1081–1086. [CrossRef]
28. Nieminen, V.; Viitanen, T.; Koski, K.; Laakso, R.; Savolainen, M. VIBFAT—Vibration-induced fatigue life estimation of the vertical tail of the F/A-18 aircraft using virtual sensing. In Proceedings of the 31st Symposium of ICAF—The International Committee on Aeronautical Fatigue and Structural Integrity, Delft, The Netherlands, 26–29 June 2023.
29. Siłka, J.; Wieczorek, M.; Woźniak, M. Recurrent neural network model for high-speed train vibration prediction from time series. *Neural Comput. Appl.* **2022**, *34*, 13305–13318. [CrossRef]
30. Jedrzykiewicz, Z.; Stojek, J.; Rosikowski, P. *Naped i Sterowanie Hydrostatyczne, Monograph*; Vist Sp. z o.o.: Łódź, Poland, 2017; p. 557.
31. Documentation of Axial Piston Pump HSP-10V Manufactured by Hydraut. Available online: https://www.hydraut.com/wp-content/uploads/2022/03/Hydraut-HSP-brochure.pdf (accessed on 10 September 2023).
32. Kluczyk, M.; Grzadziela, A. Vibration Diagnostics of the Naval Propulsion Systems. *Sci. J. Pol. Nav. Acad.* **2017**, *1*, 15–29. [CrossRef]
33. Htun, H.H.; Biehl, M.; Petkov, N. Survey of feature selection and extraction techniques for stock market prediction. *Financ. Innov.* **2023**, *9*, 26. [CrossRef] [PubMed]
34. Zulfiqar, H.; Huang, Q.L.; Lv, H.; Sun, Z.J.; Dao, F.Y.; Lin, H. Deep-4mCGP: A Deep Learning Approach to Predict 4mC Sites in Geobacter pickeringii by Using Correlation-Based Feature Selection Technique. *Int. J. Mol. Sci.* **2022**, *23*, 1251. [CrossRef] [PubMed]
35. Zebin, T.; Scully, P.; Peek, N.; Casson, A.; Ozanyan, K. Design and Implementation of a Convolutional Neural Network on an Edge Computing Smartphone for Human Activity Recognition. *IEEE Access* **2019**, *7*, 133509–133520.
36. Huang, C.; Du, J.; Nie, B.; Yu, R.; Xiong, W.; Zeng, Q. Feature Selection Method Based on Partial Least Squares and Analysis of Traditional Chinese Medicine Data. *Comput. Math. Methods Med.* **2019**, *2019*, 9580126. [CrossRef] [PubMed]
37. Thakkar, A.; Lohiya, R. Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. *Inf. Fusion* **2023**, *90*, 353–363. [CrossRef]
38. Wang, X.; Zhao, F.; Lin, P.; Chen, Y. Evaluating computing performance of deep neural network models with different backbones on IoT-based edge and cloud platforms. *Internet Things* **2022**, *20*, 100609. [CrossRef]
39. Véstias, M.P.; Duarte, R.P.; de Sousa, J.T.; Neto, H.C. Moving Deep Learning to the Edge. *Algorithms* **2020**, *13*, 125. [CrossRef]
40. Zemouri, R.; Gouriveau, R.; Zerhouni, N. Defining and applying prediction performance metrics on a recurrent NARX time series model. *Neurocomputing* **2010**, *73*, 2506–2521. [CrossRef]
41. De Gooijer, J.G.; Hyndman, R.J. 25 years of time series forecasting. *Int. J. Forecast.* **2006**, *22*, 443–473.
42. Belete, D.M.; Huchaiah, M.D. Grid search in hyperparameter optimization of machine learning models for prediction of HIV/AIDS test results. *Int. J. Comput. Appl.* **2021**, *44*, 875–886. [CrossRef]
43. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Sebastopol, CA, USA, 2016; pp. 239–242. 420–422.
44. Mao, T.; Zhou, D.X. Rates of approximation by ReLU shallow neural networks. *J. Complex.* **2023**, *79*, 101784. [CrossRef]
45. Dubey, S.R.; Singh, S.K.; Chaudhuri, B.B. Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing* **2022**, *503*, 92–108. [CrossRef]
46. Hinton, G.; Srivastava, N.; Swersky, K. RMSProp Algorithm Introduction. 2012. Available online: http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf (accessed on 6 February 2023).
47. Huber, P.J. Robust Estimation of a Location Parameter. *Ann. Math. Stat.* **1964**, *35*, 73–101. [CrossRef]
48. Kandel, I.; Castelli, M. The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. *ICT Express* **2020**, *6*, 312–315. [CrossRef]
49. Bernal, J.L.; Cummins, S.; Gasparrini, A. Interrupted time series regression for the evaluation of public health interventions: A tutorial. *Int. J. Epidemiol.* **2016**, *46*, 348–355.

50. Mukhamediev, R.I.; Symagulov, A.; Kuchin, Y.; Yakunin, K.; Yelis, M. From Classical Machine Learning to Deep Neural Networks: A Simplified Scientometric Review. *Appl. Sci.* **2021**, *11*, 5541. [CrossRef]
51. Zargar, S. *Introduction to Sequence Learning Models: RNN, LSTM, GRU*; Department of Mechanical and Aerospace Engineering, North Carolina State University: Raleigh, NC, USA, 2021.
52. Wang, Y.; Perry, M.; Whitlock, D.; Sutherland, J.W. Detecting anomalies in time series data from a manufacturing system using recurrent neural networks. *J. Manuf. Syst.* **2022**, *62*, 823–834. [CrossRef]
53. Salehinejad, H.; Baarbe, J.; Sankar, S.; Barfett, J.; Colak, E.; Valaee, S. Recent Advances in Recurrent Neural Networks. *arXiv* **2017**, arXiv:1801.01078.
54. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Comput.* **2019**, *31*, 1235–1270.
55. Hewamalage, H.; Bergmeir, C.; Bandara, K. Recurrent Neural Networks for Time Series Forecasting: Current status and future directions. *Int. J. Forecast.* **2021**, *37*, 388–427. [CrossRef]
56. Nguyen, H.; Tran, K.; Thomassey, S.; Hamad, M. Forecasting and Anomaly Detection approaches using LSTM and LSTM Autoencoder techniques with the applications in supply chain management. *Int. J. Inf. Manag.* **2021**, *57*, 102282. [CrossRef]
57. Denny Prabowo, Y.; Warnars, H.L.H.S.; Budiharto, W.; Kistijantoro, A.I.; Heryadi, Y.; Lukas. Lstm and Simple Rnn Comparison in the Problem of Sequence to Sequence on Conversation Data Using Bahasa Indonesia. In Proceedings of the 2018 Indonesian Association for Pattern Recognition International Conference (INAPR), Jakarta, India, 7–8 September 2018; pp. 51–56. [CrossRef]
58. Ganaie, M.; Hu, M.; Malik, A.; Tanveer, M.; Suganthan, P. Ensemble deep learning: A review. *Eng. Appl. Artif. Intell.* **2022**, *115*, 105151. [CrossRef]
59. Arpit, D.; Wang, H.; Zhou, Y.; Xiong, C. Ensemble of Averages: Improving Model Selection and Boosting Performance in Domain Generalization. *arXiv* **2021**, arXiv:2110.10832. https://doi.org/10.48550/arXiv.2110.10832.
60. Yao, J.; Zhang, X.; Luo, W.; Liu, C.; Ren, L. Applications of Stacking/Blending ensemble learning approaches for evaluating flash flood susceptibility. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *112*, 102932. [CrossRef]
61. Géron, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 3rd ed.; O'Reilly Media: Sebastopol, CA, USA, 2022.
62. James, G.; Witten, D.; Hastie, T.; Tibshirani, R.; Taylor, J. Linear Regression. In *An Introduction to Statistical Learning: With Applications in Python*; Springer International Publishing: Cham, Switzerland, 2023; pp. 69–134. [CrossRef]
63. Natekin, A.; Knoll, A. Gradient Boosting Machines, A Tutorial. *Front. Neurorobot.* **2013**, *7*, 21. [CrossRef] [PubMed]
64. Otchere, D.A.; Ganat, T.O.A.; Ojero, J.O.; Tackie-Otoo, B.N.; Taki, M.Y. Application of gradient boosting regression model for the evaluation of feature selection techniques in improving reservoir characterisation predictions. *J. Pet. Sci. Eng.* **2022**, *208*, 109244. [CrossRef]
65. Vassallo, D.; Vella, V.; Ellul, J. Application of Gradient Boosting Algorithms for Anti-money Laundering in Cryptocurrencies. *SN Comput. Sci.* **2021**, *2*, 143. [CrossRef]
66. Sivapalan, G.; Nundy, K.K.; Dev, S.; Cardiff, B.; John, D. ANNet: A Lightweight Neural Network for ECG Anomaly Detection in IoT Edge Sensors. *IEEE Trans. Biomed. Circuits Syst.* **2022**, *16*, 24–35. [CrossRef] [PubMed]
67. Scikit-Learn Implementation of Linear Regression Model. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html (accessed on 14 September 2023).
68. Garcia Cordero, C.; Hauke, S.; Mühlhäuser, M.; Fischer, M. Analyzing flow-based anomaly intrusion detection using Replicator Neural Networks. In Proceedings of the 2016 14th Annual Conference on Privacy, Security and Trust (PST), Auckland, New Zealand, 12–14 December 2016; Volume 12, pp. 317–324. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.