HANDS-ON APPROACH ON DEVELOPING A DEEP LEARNING ALGORITHM FOR STATE CLASSIFICATION OF A HYDRAULIC ACCUMULATOR

Oliver Mehl [1]*, Faras Brumand-Poor [2], Peter Kloft [1] , Katharina Schmitz [2]

[1] *Hydac Technology GmbH, Germany*
[2] *RWTH Aachen University, Institute for Fluid Power Drives and Systems (ifas)*
* Corresponding author: Tel.: +49 151 64421032; E-mail address: oliver.mehl@hydac.com

ABSTRACT

Hydro-pneumatic accumulators are essential components in fluid power systems, serving various purposes like dampening pulsations, stabilizing flow, and ensuring safety. Monitoring their energy state is challenging due to gas leakage, requiring knowledge of gas pressure, temperature, and volume. Real-time measurements of gas temperature and volume are difficult due to transient changes during operation. This paper introduces a novel approach to classify gas mass in bladder pressure accumulators using Deep-Learning, particularly long short-term memory (LSTM) networks. The study aims to classify load situations with minimal sensor data, providing insights into the workflow for classifying multivariate time-series data with deep neural networks. Therefore, a bladder accumulator is equipped with sensors, its dynamic behaviour is recorded and used to train and validate the LSTM network's ability to classify the gas amount inside the accumulator. This method offers a promising way to determine the pre-charge pressure, a crucial parameter for assessing the accumulator's energy state.

Keywords: Deep-Learning, Hydraulic Accumulator, Multivariate-Time-Series-Classification, Condition-Monitoring

1. INTRODUCTION

With the ongoing increase in computing power, machine learning (ML) is developing faster than ever. The field of Deep-Learning as a subcategory of ML is profiting from this trend.
In contrast to classical machine learning, deep learning algorithms are more versatile in their application, albeit more complex. Due to the computing power available in today's PCs and the continuously improved algorithms behind such deep neural networks, they can be applied to an increasing array of problems. [1]

In this work, a deep neural network is developed and trained to classify the pre-charge pressure of an active bladder pressure accumulator. Since the amount of gas contained in the accumulator is directly related to the amount of energy that can be stored by the accumulator, it is important to monitor this property to ensure the safe operation of the connected hydraulic system. The so-called pre-charge pressure p_0 describes this gas amount by indicating the pressure inside the accumulator at 20°C while the bladder fills the whole body (Figure 1). Therefore, the mass of gas can be determined by using the gas equation (1) that describes the relationship of a hypothetical ideal gas, where P is pressure, V is volume, m is the mass of the gas, R is the ideal gas constant, and T is the gas temperature. As soon as the accumulator is active, the determination of the preload pressure is not trivial anymore. Due to the high dynamics with which fluid flows into and out of the

accumulator, the volumetric flow rate and thus the volume of the gas bubble cannot be determined precisely. In addition, the temperature within the gas bubble changes just as quickly and inhomogeneously due to the rapid compression. Sufficiently accurate recording of the variables required to determine the pre-charge pressure is too complex and costly to digitize a bladder accumulator in an economically viable manner. The gas pressure can be easily and cost-effectively recorded in real-time at the valve of the bladder and the oil connection. Since the relationship between preload pressure and the associated temperature behavior is influenced by the temperature measurement's inertia, it can only be inferred from the temporal evolution of the data rather than a snapshot of the data. Hence, a single measurement at a specific time does not provide any information about the prevailing preload pressure.

$$pV = mRT \tag{1}$$

In this work, the easily acquired metrics of temperature and pressure at different points of the accumulator (Chapter 2.1) are used to train a neural network that can classify the pre-charge pressure within the reservoir with sufficient accuracy based on the characteristics of these data records considered in combination. As input for the network training the raw sensor dataSufficiently accurate in this case means to arrange the pressure in intervals of 10 bar, as this corresponds to the acceptable limit before re-charging the accumulator with gas.

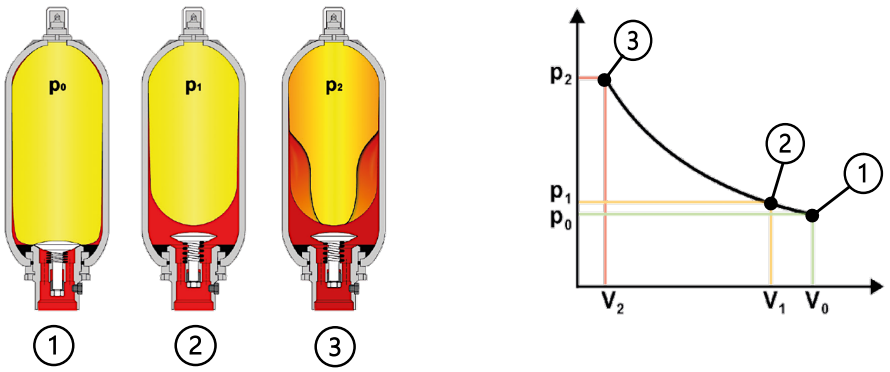

Figure 1: Overview of the states of a bladder accumulator: 1.) inactive; 2.) low operating pressure/unloaded; 3.) high operating pressure/loaded

2. METHODOLOGY

ML is finding increasing applications in the field of hydraulics, particularly in areas such as condition monitoring and predictive maintenance. Deep-Learning algorithms are increasingly being employed for these purposes. In most cases, machine learning is used to monitor entire hydraulic systems [2, 3]. Occasionally, individual components, such as hydraulic pumps, are also monitored [4].

For condition monitoring of hydraulic accumulators, ML has primarily been used to detect anomalies in operational behavior based on pressure measurements [5]. It is a novel approach in current research to equip a bladder-type hydraulic accumulator with sensors to use the data from these sensors to detect its condition using deep learning algorithms.

To accomplish tasks using ML, a structure for a data science process as shown in Figure 2 has proven useful. This process is the basis for many projects in ML. It can vary depending on the task, but the basic steps are usually the same. Since this work is also based on this procedure, the phases of such a process are presented below. [6, 7]

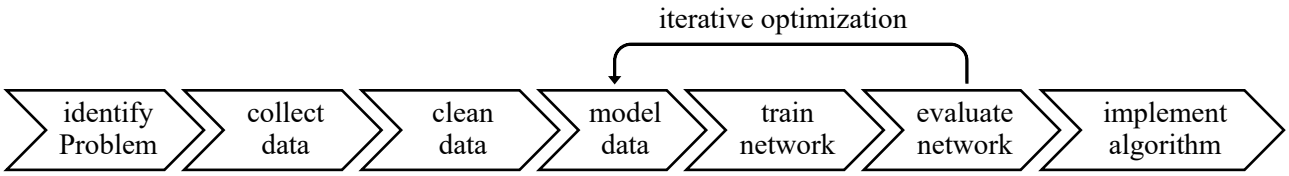

Figure 2: Steps of a typical data science process according to [6] and [7]

At the beginning of the process, the problem is identified by examining the characteristics of the data and defining the task that needs to be accomplished. Based on this analysis, decisions are made regarding the architecture and training of the neural network throughout the rest of the process. The goal is to tailor the network's structure and training approach to best address the specific problem and dataset at hand. To train the network, data is needed that represents the behavior of the pressure accumulator in different load situations at different load cycles.

2.1. Test bench setup

Therefore, a database is being recorded on the hydraulic test bench at Hydac Technology GmbH using a total of 14 Sensors mounted to a 50-liter bladder accumulator (Type SB330- 50A1/112U-330A). Figure 3 provides a more detailed overview of the kind and positioning of the used sensors. In general, the characteristic of the load being put on the accumulator at the test bench is inspired by the load on accumulators used in hydraulic plastic injection molding machines.

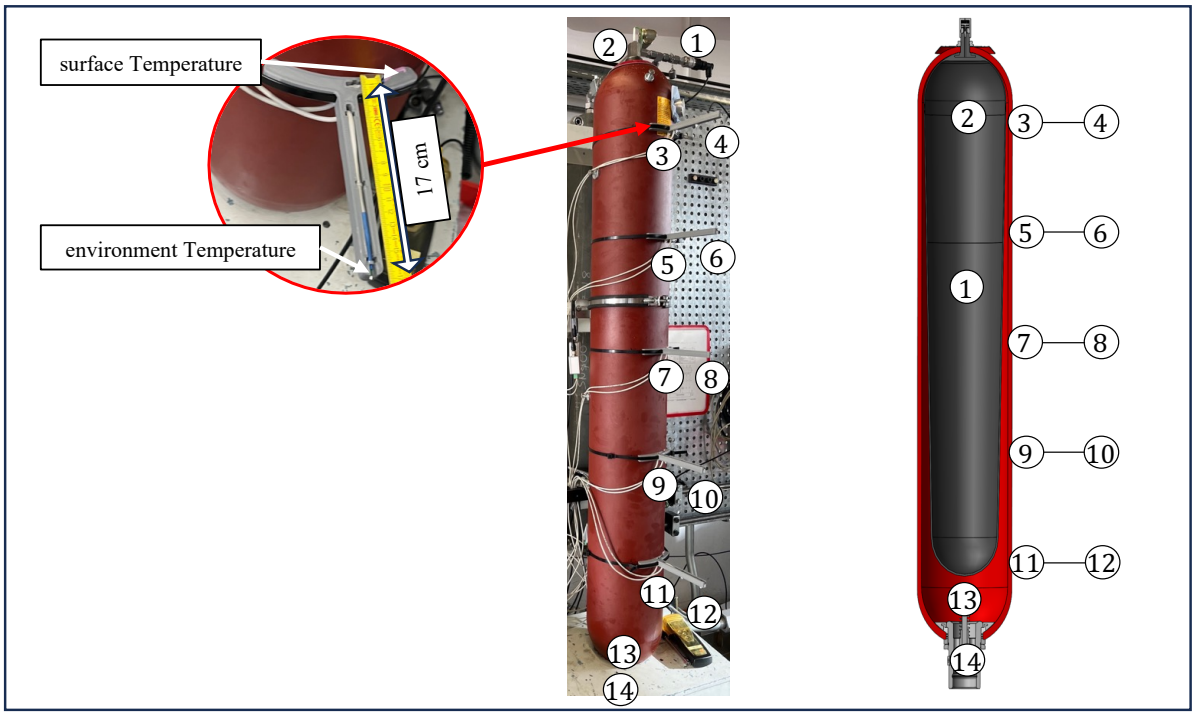

Figure 3: Detailed view of temperature sensor mount (left); picture of test bench with marked sensors (middle); schematic view of sensor positioning (right)

The sensors with the numbers 1 and 13 in Figure 3 are pressure sensors of type HDA4800 from Hydac Electronics GmbH. Number 2 and 14 are sheathed thermocouples type T5 from Conatex GmbH where the thermocouple on the gas side (nr. 2) has a nominal diameter of 0.5 mm and the one on the oil side of the accumulator (nr. 14) 2 mm. The sensors 3-12 are PT1000 temperature sensors where number 3,5,7,9 and 11 are reading the surface temperature of the accumulator and the sensors 4,6,8,10 and 12 read the corresponding environment temperature.

These temperature sensors are positioned in pairs along the accumulator. Longitudinal positioning is essential because hydraulic bladder accumulators exhibit a characteristic longitudinal heat distribution during usage. This distribution results from the bladder's varying contact with the pressure tank of the accumulator, depending on the loading state. The arrangement in pairs is intended to measure the temperature gradient at specific points on the accumulator, which is proportional to the heat flow from the surface to the surrounding environment at those locations.

It's worth noting that other sensors have been considered for data recording, such as vibration sensors to detect changes in resonance behavior, radar sensors to track the shape and behavior of the gas bladder, or an infrared pyrometer to measure the gas temperature. Ultimately, these sensors were not utilized due to challenges in recording and managing the substantial amounts of data generated by vibration recording, as well as high hardware costs.

2.2. Recorded data

For the neural network to learn the system behavior of the pressure accumulator, the accumulator is exposed to a total of three different pressure profiles with different variations. For the variations in the pressure profiles, different holding times are set at various points of the profiles, or the pressure at which the curve runs is varied. A total of 11 different pressure cycles were generated through these variations. Each of these cycles was recorded over several hours at various pre-set preload pressures. Figure 4 provides an overview of the recorded database with examples of the contained cycles. The shown table of the database indicates which of the 11 cycles has been recorded at which preload pressure. All 14 sensors were recorded at a sampling rate of 100 Hz by two CMU1000 devices from Hydac Electronics GmbH that have been synced in time. As part of the data cleaning step (Figure 2), faulty measurement outliers were corrected, and missing measurements were supplemented through interpolation. In addition, two types of labels were added to the data: Firstly, the data is labeled with the preload pressure that prevailed during the recording, and secondly, a status label is added indicating whether the accumulator was 'inactive', 'good', or 'heating' at the time of recording. With this recorded, preprocessed, and labeled database, the next step is to train the neural network.

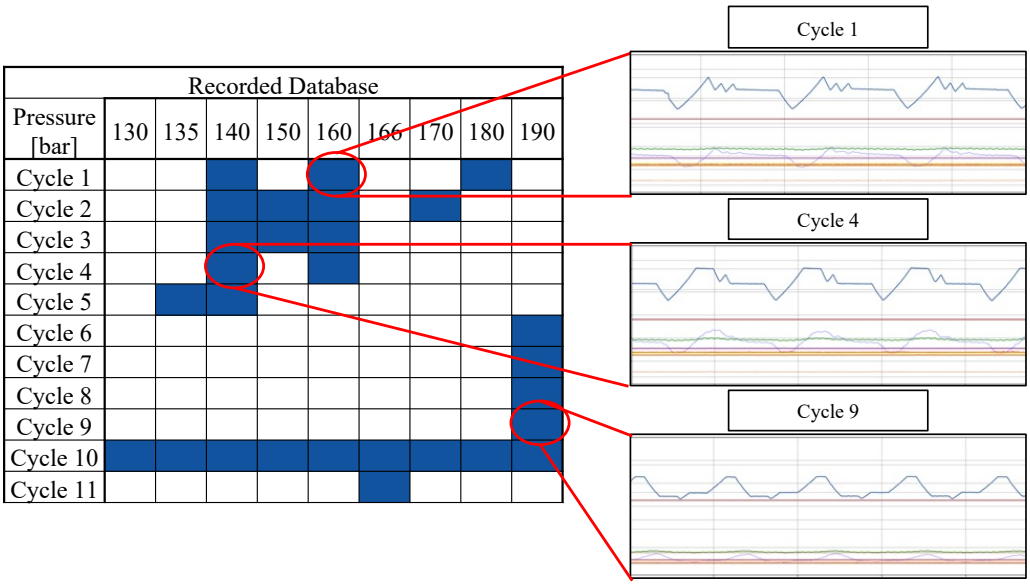

Figure 4: Overview of recorded database (left) with examples of different cycles and their different pressure profiles (right)

2.3. Network training

The basic structure of this training consists of a data pipeline that processes and models the data from the database and feeds it into the network. Figure 5 provides an overview of the recorded data of one day with the labels that have been given during pre-processing. The data is prepared in such a manner that the network can effectively recognize and learn relationships between the recorded sensor data and the preload pressure provided as the solution. Both the data pipeline and the neural network can be configured with a variety of parameters. The following subsections explain the structure of the data pipeline, the connected network, and their configurations. Subsequently, the approach to finding an ideal configuration will be explained.

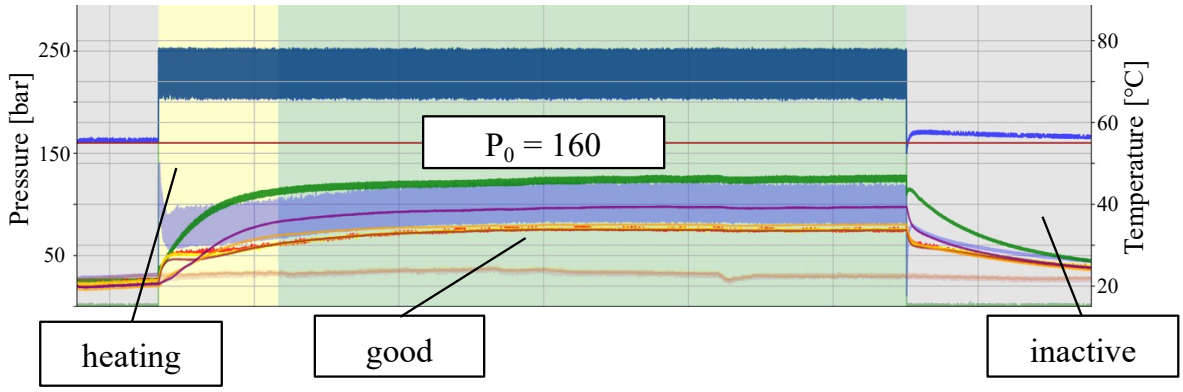

Figure 5: Preprocessed and labeled data of one day. Each cycle at each set pre-load pressure has been recorded for one day.

Data-Pipeline

The pipeline processes the data from the database in eight consecutive steps, allowing the network to recognize and learn the relationships between the sensor data patterns and the preload pressure.

1. The data to be used for training is read from the database and loaded into Pandas DataFrames [8].
2. The data is divided into time windows and organized into batches. This division into time windows is a common practice in processing time-series data to provide the network with the information contained in the temporal evolution [9].
3. Faulty batches, undesired time ranges, and values that should not be considered are filtered.
4. The data sampling rate is reduced to reduce computational load, by considering only every n-th data point for network training. Where n is an adjustable parameter to configure the desired sampling rate that is processed by the network.
5. The batches are normalized based on predefined threshold values to ensure consistent processing. These threshold values are derived from the product data sheet of the used accumulator.
6. The batches are divided into training, validation, and testing sets while preserving the chronological order.
7. The preload pressure is categorized into intervals of 10 bar steps to enable classification.
8. Finally, the data is transformed into the required format for the network of Numpy-Arrays [10].

In this framework, several variable factors influence how the network learns from the data. In this work, the number of sensors, the window size, and the sampling rate are examined in more detail.

2.4. Machine Learning Model

As mentioned in the introduction, a deep neural network is trained to autonomously recognize and learn features from the data. This entails connecting multiple layers of neurons. The type of neurons in these layers and how they are interconnected depends on the characteristics of the data and the task at hand. For the layers in this study, a neural network consisting of LSTM layers is constructed. These models are specifically designed to capture relationships over multiple time steps without consuming excessive hardware memory resources [11].

The neural network trained in this study consists of three consecutive LSTM layers, each with 64 neurons, followed by three fully connected dense layers. The subsequent dense layers have 64, 32, and, 8 neurons, which is a common approach to reduce the features identified and processed by the LSTM layers to match the number of categories to be classified. The number of units in the last layer must match the number of possible solutions. This described structure has been identified and tested in a heuristic approach. Table 2 provides a summary of the basic configuration of the network. In addition to the basic architecture, several other settings influence the learning behavior of the neural network.

Table 2: Default parameters of data pipeline and neural network that are tested during network training.

Data Pipeline	
Parameter	Value
Considered sample rate [Hz]	50
Time window size [s]	60
Neural network	
Learning rate	0.001
Dropout rate	0
Number of processed batches	128

Some settings, such as the activation functions of the Dense-Layers (set to Rectified Linear Unit) and the "return sequences" setting of the LSTM layers (set to “true”), are fixed because they have proven effective for time data classification tasks. However, settings like the dropout rate, batch size, and learning rate are randomly chosen (or set to their default values) and will be analyzed during the hyperparameter tuning process to improve training performance.

2.5. Training Process

The training of the network is designed to reveal which information, in what shape, is best suited for the intended classification task. At the same time, it aims to provide a clear understanding of how well the network learns with the chosen data pipeline and network settings. After a basic configuration has been successfully tested, achieving a classification accuracy of 99% for cycles that are included in the training data (known cycles), the focus shifts to evaluating how well the network can classify new cycles that are not contained in the training data. Considering the amount and diversity of recorded data, it is unlikely that the network, after being trained on this data, can classify significantly divergent cycles effectively. Furthermore, it is interesting to identify which sensors are relevant for a reliable classification of the accumulator's state.
To determine the optimal configuration of the pipeline and network, the training process is divided into three steps:

Firstly, in the hyperparameter tuning phase, the configuration parameters defined as variables are examined for their impact on training. This involves exploring different values for these parameters to find the combinations that yield the best results. The training process for this first step is as follows: For each parameter to be tested, neural networks are trained with different-sized datasets. These datasets contain training-, validation- and test-data. Therefore, the trained networks are already tested on the cycles they have been trained on. Each of these trained networks is then tested on the remaining, yet for the network unknown, data from the database. This unknown data contains all the varied cycles with differing characteristics compared to the training dataset.
This involves a form of cross-validation, with the particularity that, after training, the performance is tested on a dataset derived from the training data, and the final performance evaluation is separately conducted using a dataset distinct from the training data. By gradually expanding the dataset, a correct interaction between data pipeline and neural network can be assured, since the learning behaviour is expected to improve with an increasing dataset. This approach allows for a systematic exploration of how different configurations and data sizes impact the network's performance. It helps in understanding which parameter settings led to better generalization and robustness, as the network is tested on unseen data, thus providing insights into the model's ability to handle new and diverse samples. The goal is to find the optimal combination of parameters that maximizes the network's classification accuracy and generalization ability.
Secondly, in the network optimization step, the combination of the most promising parameters from the hyperparameter tuning phase is fine-tuned. Since the parameters, that have been individually tested in the hyperparameter tuning, can influence each other, this step aims to find the best combined setting of parameters. By breaking down the training process into these two steps, the study aims to systematically explore the impact of different configurations and identify the most effective settings to achieve the desired classification performance.
In the final third step, after evaluating the importance of each sensor for classification, the best configuration is tested with the data of fewer sensors. This aims to determine the number of sensors that contain significant information about the state of the accumulator and therefore are needed for a robust classification. Since more sensors also mean more data to be processed it is desirable to have as few sensors as possible to reduce the cost for hardware regarding not only the sensors themselves but also the necessary hardware for processing and storing the data.

2.6. Evaluation

The evaluation of the trained networks is conducted by analyzing the training metrics (progression of loss and accuracy) of the network training and the test performance on classifying unseen data. When considering the training metrics, an assessment of the training behavior’s stability is performed. The evaluation of test performance provides insights into how effectively the analyzed network can classify data that differs from the training data and therefore how general the network can model the real world behaviour of the accumulator based on the given sensor data.

In the analysis of training metrics, the loss and accuracy values are observed across the epochs of the conducted training. For this work, the categorical-crossentropy-function was employed. Observing these graphs can determine whether the network has reached an optimum and, if so, how rapid and reliable [12]. Ideally, in this scenario, these metrics approach their optimum values (accuracy towards one and loss towards zero) quickly at the beginning and then level off towards the end of training. Furthermore, validation performance should closely follow the training values. If the trajectory of these metrics is highly volatile or fails to converge to an optimum, this indicates that the network might struggle to learn the relationships between the input data and the class labels it is meant to predict (as illustrated in Figure 6).

To evaluate the test performance, the network to be analyzed must classify all cycles that were not used for training. For this classification, the data is processed by the data pipeline in the same manner as during training. The test data is fed to the trained network and the predicted results are then compared with the corresponding labels. The percentage of correctly classified batches is used as a metric to measure transfer performance.

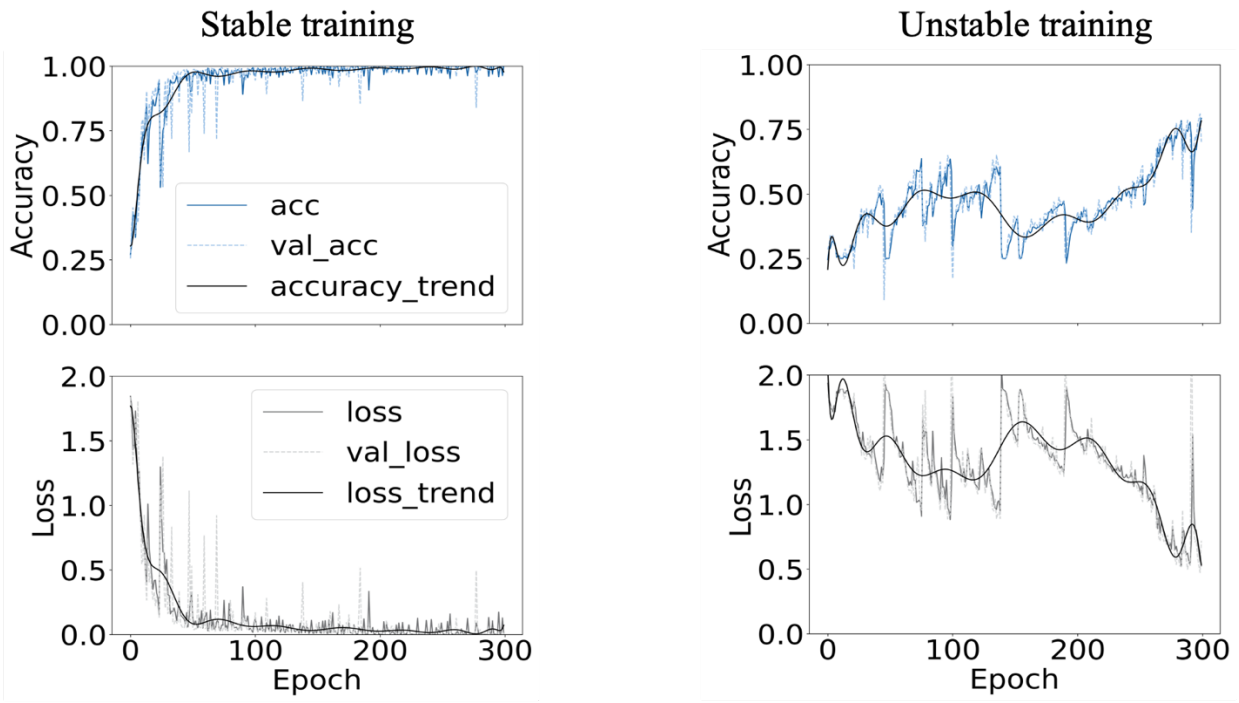

Figure 6: Comparison of examples of stable training metrics (left) and unstable training metrics (right).

As mentioned, every sensor's importance for the classification should be analyzed. It is then tested by training neural networks without the sensors that provide the least amount of information according to the before-determined importance. Since the data of each sensor is being processed as one feature by the pipeline as well as the first network layer, a feature importance analysis must be performed. One way to accomplish this is the feature permutation analysis [14], which is also being used in this work. The concept behind this permutation analysis is to permute the data of one feature (in this case one sensor) to delete information that is contained in the pattern of the sensor data and let the trained network classify the biased data. The drop in accuracy of the classification after the permutation indicates how important the information of that sensor is for the classification. In this work, the permutation is being performed in two different ways. One method is to take the data of one sensor and replace the data of each batch with the average value of that batch. That way the information contained in the pattern is replaced by a flat line. The second method is to shuffle the values of one sensor in each batch that is classified so that the pattern does not follow any characteristic curve anymore. Although both approaches yield similar results, in this work, the second (shuffle) method was used for evaluation, as the neural network appears to respond more sensitively to it.

3. RESULTS AND DISCUSSION

In the hyperparameter tuning phase, starting from the successfully tested base configuration, the parameters summarized in Table 2 were individually varied and analyzed. As a result, a total of 12 neural networks were trained, each using increasingly diverse and varying dataset sizes. These networks were then compared based on the presented metrics.

As expected, the learning behavior of the network becomes increasingly stable with larger and more diverse datasets. This behavior is characteristic of deep neural networks and indicates a successful synergy between the data pipeline and the neural architecture [15]. Furthermore, it can be observed that nearly every tested configuration is capable of correctly classifying known cycles with an accuracy of 99%. This consistency in achieving high classification accuracy highlights the effectiveness of the network configurations in capturing the underlying patterns in the data. Based on these findings, the determination of which settings to use for each of the 5 adjustable parameters in the final configuration is primarily guided by the described test performance metric of the networks trained on the largest dataset. However, considerations such as available computational capacity and the stability of the learning behavior also contribute to the selection of parameters like the sampling rate, window size, and batch size. The settings for each parameter that reached the best results are then put together into a compiled configuration. The network trained with this compiled configuration reaches an accuracy on the test data of 53%.

This compiled configuration is then tested for additional improvement potential during the network optimization phase. Since individual parameters can potentially influence each other, this iterative process aims to fine-tune the network by exploring possible synergies between different parameter settings. The goal here is to achieve the highest possible classification accuracy and robustness while effectively utilizing available resources. The tested variations with their results of the optimization phase can be found in Table 3. Subsequently, in this step, the test accuracy could be increased up to 70%. That means, that 70% of batches of new cycles have been classified correctly by the neural network.

Table 3: Overview of the trained neural networks

Window size [s]	Sample rate [Hz]	Dropout rate [%]	Nr. of processed batches	Learning rate	Sensor nr. according to Figure 3	Test-accuracy [%]
Base configuration before hyperparameter tuning						
60	50	0	128	0.001	all	**36**
Combined configuration after hyperparameter tuning						
15	50	40	200	0.001	all	**53**
Tested configurations during network optimization						
15	16	40	200	0.005	all	**47**
15	16	40	200	0.001	all	**59**
15	50	40	200	0.005	all	**70**
Results of training with a reduced number of sensors						
15	50	40	200	0.005	1,2,3,5,6,9,13,14	**65**
15	50	40	200	0.005	1,2,3,10,13,14	**56**

In the last step, this final network is used to perform the described feature importance analysis to identify important sensors. This analysis indicates that most of the environment temperature sensors do not contribute a lot of information for the classification task, while the surface temperature sensors as well as the pressure and temperature sensors in oil and gas side of the accumulator seem to deliver relevant data. To validate this outcome, two additional neural networks are trained. One uses data from eight sensors, and another utilizes data from six sensors. The results of these two network trainings are summarized in Table 3. The network trained with data from eight sensors managed to achieve a transfer performance of 65% accuracy. This is only a 5% decrease compared to the network trained using data from all 14 sensors. On the other hand, training with the specified six sensors resulted in a transfer performance accuracy of 56%. When additionally comparing the training metrics of the three neural networks trained with data from all sensors, eight sensors, and six sensors (Figure 7), it also becomes evident that the network's learning behavior is getting more uncertain as fewer sensor data are available. The observation of generalization capabilities and training metrics imply that the sensors used provide information about the energetic state of the accumulator and that the temperature distribution along the accumulator's surface provides significant value for predicting pre-load pressure. Hence, for the final tests, the configuration using the mentioned 8 sensors will be employed. This configuration strikes a favorable balance between a reduced number of sensors and delivering strong results in classifying unknown cycles.

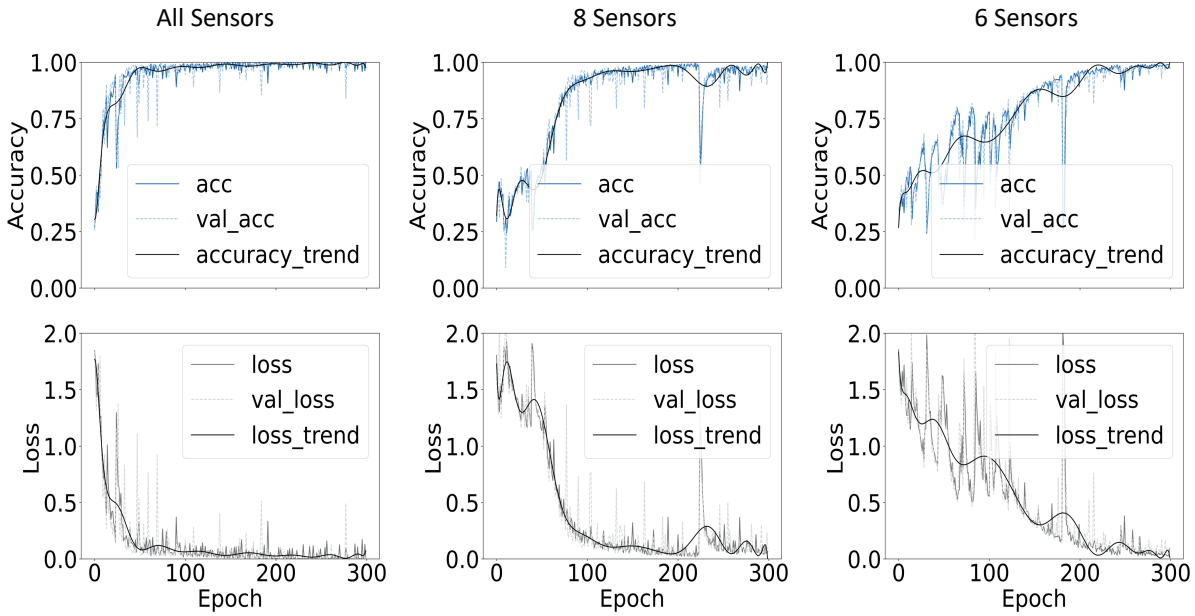

Figure 7: Comparision of the training metrics with decreasing number of features/sensor readings.

4. TESTING THE FINAL CONFIGURATION

In the previous stages of this work, around 50% of the data contained in the dataset was used not for training but for subsequent evaluation of test performance. The final assessment of the potential of this network structure will involve training a neural network using as much of the available data as possible. During the hyperparameter tuning phase, it has become evident that both the stability of learning behavior and the transfer performance of the trained networks notably increase with a growing training dataset. Therefore, during this final test, it is expected that the network's transfer performance will further improve, as the dataset becomes significantly larger and encompasses a greater diversity of different pressure profiles. This final training aims to best replicate the scenario where the pressure accumulator is connected in a new environment, and the operational data of the first day, including the target preload pressure, is recorded and contained in the training data. The

network is trained on all recorded cycles with all preload pressures except for cycle 3, of which only the data at 160 bar preload pressure is included. This network is then tested using the remaining data from cycle 3, which involves preload pressures different from the target. This test assesses how well the network can classify a preload pressure deviating from the intended one.
Due to the larger dataset, the algorithm in the final configuration is trained for 500 epochs. The evaluation demonstrates that the network training progresses very stably. Furthermore, the network managed to correctly classify about 98% of the previously unseen data from cycle 3, signifying an excellent test performance. These results suggest that the trained network is adept at generalizing its learned patterns to scenarios beyond its training data. This provides a strong indication of the network's capability to effectively handle various real-world situations involving differing preload pressures.

Finally, the network is tasked with classifying the data from an entirely new cycle. The pressure profile follows a sawtooth pattern with a labeled preload pressure of 160 bar, significantly differing from all the cycles present in the training dataset. In predicting the preload pressure of this unknown profile, the neural network correctly classifies approximately 71% of the segmented time windows.

5. CONCLUSION

In summary, it can be concluded that the developed neural network can classify preload pressure conditions it has learned with high accuracy and confidence. The closer the pressure profile being classified aligns with the learned training data, the more reliable the classification becomes. The hyperparameter tuning carried out for network development has shown promising results in enhancing the network's performance. Due to the multitude of settings available in data preprocessing and network configuration, this work did not achieve a global optimum in training configuration, indicating further optimization potential. Furthermore, it is evident that prediction quality and the ability to classify unfamiliar conditions significantly improve when more data from diverse load scenarios are available for training. This underscores the need for additional, diverse operational data on the pressure accumulator.

The conducted tests of the finally trained neural network also reveal additional potentials to further enhance the robustness of predictions. During the evaluation of feature importance, it is evident that the network can withstand the failure of individual sensors and still classify the state of the accumulator accurately. The state classification can be further optimized by considering the classification of the last assessed time windows. In the presented model, only the current time window is classified in isolation. As the system's condition changes gradually, multiple consecutive time windows can be collectively considered for state assessment. By implementing simple logical operations at the software level using these methods, the robustness and accuracy of predictions can be significantly improved without the need to train a new neural network.

This enhanced network performance in conjunction with a more diverse and expansive dataset reflects the benefit of using deep neural networks (referred to as deep learning). Another advantage of this network structure lies in the fact that the combination of data pre-processing and the neural network is fundamentally suited for processing such sensor data and subsequently automating the recognition of relationships between data and labels. This stems from the capability of such neural networks to autonomously identify connections between data and the categories they are meant to classify.

NOMENCLATURE

p	Pressure	bar
V	Volume	m^3
m	Mass	kg
R	Gas constant	$J \cdot mol^{-1} \cdot K^{-1}$
T	Temperature	K
p_0	Pre-charge pressure at 20°C	bar

REFERENCES

[1] I. Goodfellw, Deep learning. eng. Adaptive computation and machine learning series, Massachusetts: MIT Press, 2016.

[2] N. Helwig, E. Pignanelli and A. Schütze, "Condition monitoring of a complex hydraulic system using multivariate statistics," 2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings, 2015.

[3] K. Huang, S. Wu, F. Li, C. Yang and W. Gui, "Fault Diagnosis of Hydraulic Systems Based on Deep Learning Model With Multirate Data Samples," IEEE Transactions on Neural Networks and Learning Systems, no. 33, 2022.

[4] S. Tang, Y. Zhu and S. Yuan, "An adaptive deep learning model towards fault diagnosis of hydraulic piston pump using pressure signal," Engineering Failure Analysis, 01.08.2022.

[5] M.-H. Park, S. Chakraborty, Q. D. Vuong, D.-H. Noh, J.-W. Lee, J.-U. Lee, J.-H. Choi and W.-J. Lee, "Anomaly Detection Based on Time Series Data of Hydraulic Accumulator," Sensors, 01 2022.

[6] A. Kishore, "A Layman's Guide to Data Science Workflow," 2023. [Online]. Available: https://www.knowledgehut.com/blog/data- science/data-science-workflow. [Accessed 25.10.2023].

[7] S. Salimon, "Data Science Workflow: How to Create and Structure it Simplified 101," 2023. [Online]. Available: https: //hevodata.com/learn/data-science-workflows/. [Accessed 25.10.2023].

[8] W. McKinney, "Documentation Pandas," 2023. [Online]. Available: https://pandas.pydata.org/docs/. [Accessed 25.10.2023].

[9] M. M. Rahman, Multivariate Time Series Classification of Sensor Data from an Industrial Drying Hopper: A Deep Learning Approach, West Virginia: West Virginia University Libraries, 2021.

[10] "Documentation Numpy," [Online]. Available: https://numpy.org/doc/stable/index.html. [Accessed 25.10.2023].

[11] J. Schmidhuber and S. Hochreiter, "Long Short-term Memory," Neural computation, no. 9, 1997.

[12] " Artificial Intelligence Wiki," 2020. [Online]. Available: https://machine-learning.paperspace.com/wiki/accuracy-and-loss. [Accessed 25.10.2023].

[13] "Keras documentation," [Online]. Available: https://keras.io/about/. [Accessed 25.10.2023].

[14] S. Raschka, "Youtube - Feature Permutation Importance," 2022. [Online]. Available: https://www.youtube.com/watch?v=VUvShOEFdQo. [Accessed 25.10.2023].

[15] D. Sarkar, R. Bali and T. Sharma, Practical Machine Learning with Python, Berkeley, CA: Springer, 2018.