TUM Data Innovation Lab

Munich Data Science Institute (MDSI)

Technical University of Munich

&

IfTA GmbH

Final report of project:

Support the Energy Transition - Monitoring Hydro Turbines

Authors	Paul Faschingbauer, Michael Bernardi, Marsel Kolovski, Jasmina Stratorska, Hammad Basit
Mentor(s)	M.Sc. Fabian Legl, Dr. Driek Rouwenhorst
Co-Mentor	M.Sc. Cristina Cipriani
Project Lead	Dr. Ricardo Acevedo Cabra (MDSI)
Supervisor	Prof. Dr. Massimo Fornasier (MDSI)

Jul 2023

Abstract

Pumped storage hydroelectric plants are becoming increasingly relevant due to the current transition towards renewable energy. Regular maintenance must diagnose and rectify issues on time to support the continual operation and maximise the energy output of these plants. Mechanical components in these systems are vulnerable to wear from various sources, including friction, erosion, and cavitation. This can lead to issues such as imbalances in rotating components, further degrading parts at an accelerating rate.
Working in collaboration with IfTA GmbH on datasets provided by Kraftwerke Oberhasli AG (KWO), we aim to utilise machine learning methods to help automate the diagnosis of mechanical issues in the machinery. A typical method to achieve this is to train a machine learning model on a dataset when the system is running in the desired normal state. Once sufficiently trained, one can apply the model to new data and compare its predictions against the observed true values. This process is also called anomaly detection and is the primary goal of this project.
In our case, KWO has graciously provided us with two comprehensive datasets, which are used as the foundation for our ML-based diagnostic system. We use operational data such as water pressures, electrical load, and RPM from one dataset to predict machine vibrations in the other. Discrepancies between true and predicted values suggest that the machinery may no longer be in the normal state, which can be the basis for diagnosing mechanical issues.
This report explores different approaches to training such models, from the preparation of data to the comparisons of the predicted and true data. We investigate using many different models, including linear regression, non-temporal neural networks, recurrent neural networks, and auto-encoders. Finally, we create an anomaly detection system and conduct a blind test against a dataset with artificial anomalies constructed by IfTA.

Contents

1 Introduction

1.1 Background

For the purpose of sustainable energy production, renewable energy is becoming more and more important. One major challenge is the so-called "duck curve"[1] which refers to the way that peak renewable energy production is not synchronous with peak demand. Pumped hydroelectric storage can mitigate this issue as it provides the capability to store the output from renewable sources to be later used during peak demand[2]. This is accomplished by using a pump to pump water uphill and then later using a turbine to recuperate the energy as the water flows back downhill[5].
The subject of this report is the "Kraftwerke Oberhasli AG, PSW Grimsel 2, Machine Group 1" located in the Swiss Alps in the canton of Bern. It can pump water from the lower "Grimsel" reservoir to the upper "Oberaar" reservoir and reverse the process to generate power.

1.2 Goals

The motivation for this project comes from the need to effectively monitor and detect anomalies in machine operation modes in renewable energy systems. By developing a system that can find anomalies based on the unique vibration patterns of each mode, we aim to prevent damage, reduce machine downtime, and make renewable energy sources more reliable. These goals align with the ongoing efforts to optimise the performance and sustainability of renewable energy systems, contributing to the development of clean and efficient energy solutions.
In the field of vibrational analysis, machines vibrate differently depending on the conditions of operation. Each condition has its own unique "vibration fingerprint" that depends on various parameters such as load and rotational speed. By using machine learning, we can train a model to connect the vibration data with the corresponding condition. If the model's predicted vibrations don't match the real-time vibration data, it means there might be a problem or something different from what we expect.
An automated anomaly detection system has the potential to support the following efforts:

1. **Early Recognition and Prevention of Damage:** By finding anomalies in the vibration patterns, we can detect potential problems early. This way, actions can be taken and damages can be fixed before they become serious, reducing the risk of large and costly failures.

2. **Less Machine Downtime:** By addressing possible problems early on, we can also reduce the time machines are out of service. This makes renewable energy systems more efficient and productive, ensuring they can generate power without interruptions.

3. **Maintenance Based on Monitoring:** With a good anomaly detection system, we can plan maintenance based on how the machine is actually doing. This helps us use our resources efficiently and avoid unnecessary maintenance work.

4. **Improved Lifespan:** With improved anomaly detection leading to improved maintenance, we make the machines work better, potentially extending their lifespan. This increases the overall reliability and cost-effectiveness of renewable energy systems in the long term.

2 Data Acquisition & Datasets

Kraftwerke Oberhasli AG (KWO) provides us with two datasets focusing on one of the four machine groups at the site. The first is a 'rotordynamic dataset' measured by an IfTA system and the second contains other operational data about the pump and turbine (machine group) in question.
The rotordynamic dataset provides data on shaft displacement both laterally and axially, as well as bearing housing vibrations and RPM (rotations per minute).
The second dataset is an operational dataset that contains information about the surrounding system. It covers various parameters such as flow rates, the guide vane position, pressure in the penstocks/manifolds, produced or consumed power, and oil and cooling water temperatures. Additionally, the dataset includes water levels in the reservoirs and surge tanks.
By combining these datasets, a comprehensive understanding of the machine's performance and its relationship with the surrounding system can be obtained. This information is valuable for analysing and optimising the operation of the machine group in terms of efficiency, reliability, and overall system health.

2.1 Rotordynamic Dataset

The rotordynamic dataset provides detailed measurements at both the pump and the turbine bearings. At each bearing, there are two sensors positioned 90° apart, which measure the displacement of the shaft in two dimensions. This allows for a comprehensive record of the shaft's movement and positioning. The measurements and positions of the sensors are compliant to the international DIN-ISO 20816-52018 standard.
Furthermore, the dataset includes a keyphasor sensor that enables the accurate measurement of the shaft's rotational speed (RPM). This information is crucial for analysing oscillations in the motion in terms of rotational speed. For example, a large oscillation synchronised with the rotational speed can indicate an imbalance in the shaft. A faster oscillation at a frequency of 17 times the rotational speed can indicate a fixed disturbance in the flow. This is because there are 17 blades in the turbine and each blade will pass the disturbance 17 times per shaft rotation. Figure 1 depicts the layout of the keyphasor and aforementioned shaft displacement sensors.
In addition to these sensors, the rotordynamic includes two accelerometers per bearing that capture vibrations of the bearing housing. These accelerometers provide valuable insights into vibrational energy dissipating into the surrounding structure.
Lastly, there is a sensor dedicated to measuring axial displacement. Combining this with the other data allows us to recreate the complete motion of the shaft in all three dimensions.

This dataset consists of a total of 10 sensors, including those for measuring shaft displacement, bearing housing vibrations, RPM, and axial displacement. These measurements allow us to accurately record the 3D-position of the shaft at the pump and turbine bearings, and also track the vibrational energy dissipating through these bearing housings. Figure 2 shows a conceptual layout of the shaft and two bearings.

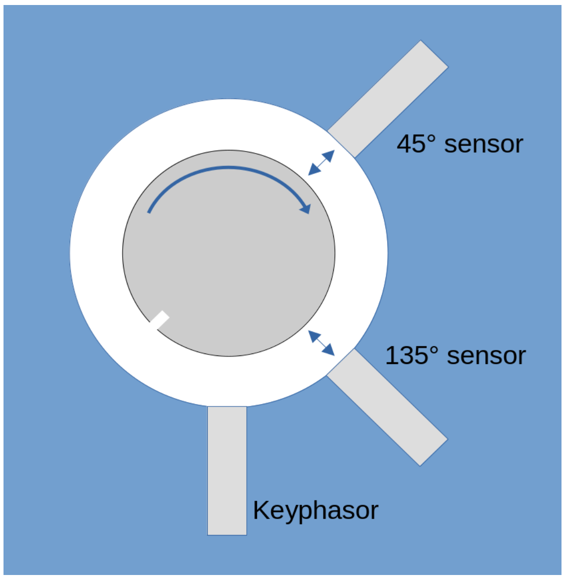


Figure 1: Conceptional cross-sectional diagram depicting the sensors measuring the shaft.

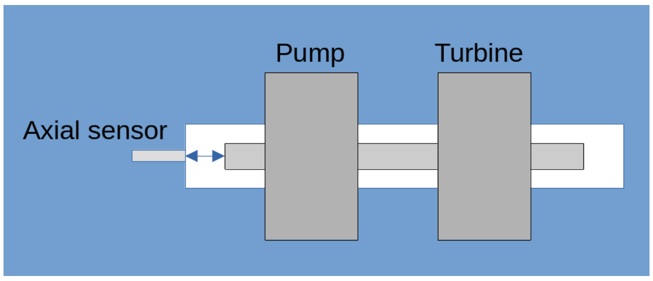


Figure 2: Conceptional diagram depicting the layout of the pump and turbine on the same shaft. Bearing and shaft sensors are not depicted.

The raw data is recorded at a very high frequency, and so we only work with processed aggregates of this data. A Fourier transformation is used to aggregate the motion of the shaft over the course of approximately 600 milliseconds. The frequency data is recorded for 7 frequency windows, which correlate to the harmonics of the shaft rotation frequency. We can use this to recreate an orbit plot of the average motion of the shaft over the time period, as seen in figure 3. Other aggregates important for the shaft and bearing housing vibrations are also stored, such as the mean, peak-to-peak value, and RMS (root mean square). In the case of RMS, this is also aggregated for each of the 7 frequency windows.

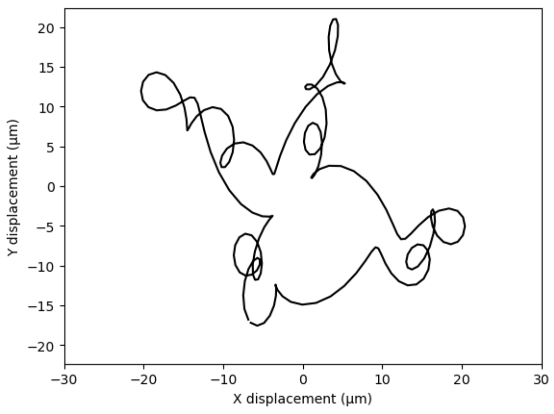


Figure 3: Example of an orbit plot depicting the 2 dimensional motion of the shaft at the pump.

2.2 Operational Dataset

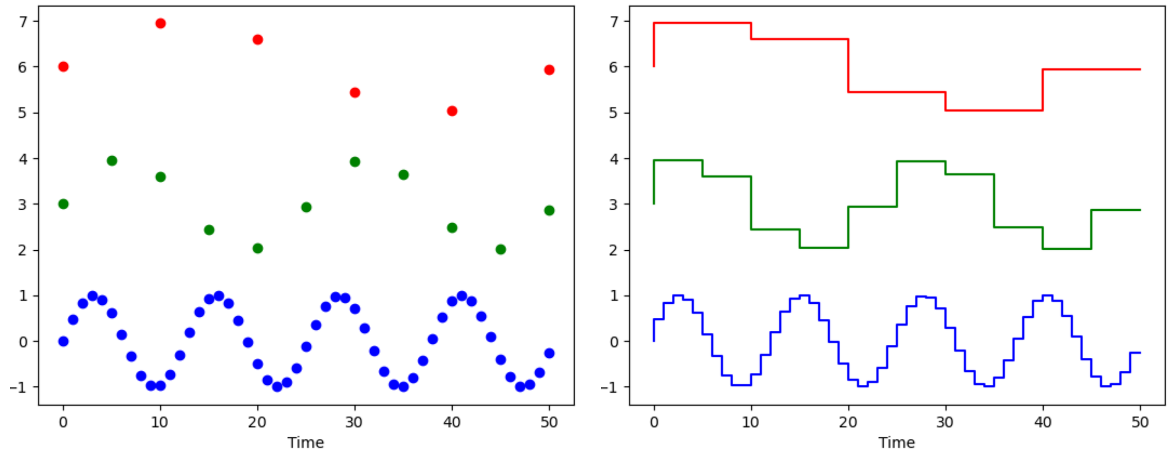
The operational dataset is organised into three subsets, each providing valuable information about the system's operation. These subsets are listed below in order of importance:

- **Operational Data at the Machine Itself:** This subset includes critical parameters directly recorded at the machine. It encompasses measurements such as flow rates and water pressures within the manifolds, the power produced or consumed, reactive power, and the position of the turbine guide vanes.
- **Temperature of Machine Components and Fluids:** This subset focuses on monitoring the temperature of various machine components and fluids. It includes measurements of oil, water, and air temperatures, providing insights into the thermal conditions of the system. Furthermore, temperatures of generator coils and stators, seals, and bearings are recorded. These temperature measurements are crucial for assessing heat management and provide a proxy for the physical condition of the bearings and the motor-generator.
- **Water Levels in the Lakes and Chambers:** The final subset of the operational dataset pertains to water levels within the system. It includes measurements of lake water levels which can affect pressures at the machine. Additionally, water levels in the pipes and surge tank are monitored to assess the hydraulic balance and water flow throughout the system.

2.3 Data Preparation

Merging

To create a model we must first combine the data we have by merging all data from the operational and rotordynamic datasets into a single dataset. Since our data comes from sensors in real time, each data point is indexed by a timestamp. However, not all sensors are measuring synchronously or with the same frequency. The sensors record data



(a) Discrete signal (b) Interpolated signal

Figure 4: Signal resampling example.

at discrete points in time and thus we have gaps in the data between the data points generated by different sensors. This is depicted in figure 4a.
To merge the different components of the operational data, we must resample the lower-frequency streams at the higher frequency and interpolate the missing values. We experiment with various interpolation techniques and notice no appreciable difference in results, so we settle on using a simple back-filling technique which imputes the future observation to earlier points. This is depicted in figure 4b.
The resampling process also truncates the timestamps to the nearest second. For example the timestamp `12:20:17.334` becomes `12:20:17`. When multiple observations are truncated to the same second, we take the mean. We repeat this resampling process when merging the operational data with the rotordynamic data. This ensures we can join on the timestamp field, otherwise data from the two different datasets will rarely join due to small differences in the milliseconds field.

Filtering

After simplifying the dependent variables by reducing them to a single one, we analyse the dataset to filter out outliers. Initially, we train our model with good data from situations where the machine is in continuous operation. This presents the easiest scenario for the model to learn the correlations between input and output datasets. Like all rotating machinery, during startup and shutdown, the machine spins slower and is likely to experience larger vibrations that are not typical during normal operation, and this can be challenging to capture in a basic model. We can see in figure 5 that in its normal state the rotational speed of the shaft is around 700-750 RPM. It has a bimodal distribution due to the two pumping and generating modes of operation. Although not depicted in the figure, there is also a lot of data where the machine is stopped. Therefore we decide to filter out data where the machine is not spinning faster than 600 RPM. We also filter out cases where the power production/consumption is zero, as this can occur in some erroneous rows in the dataset.
After some analysis, we also deduce that some of the water level variables have to be

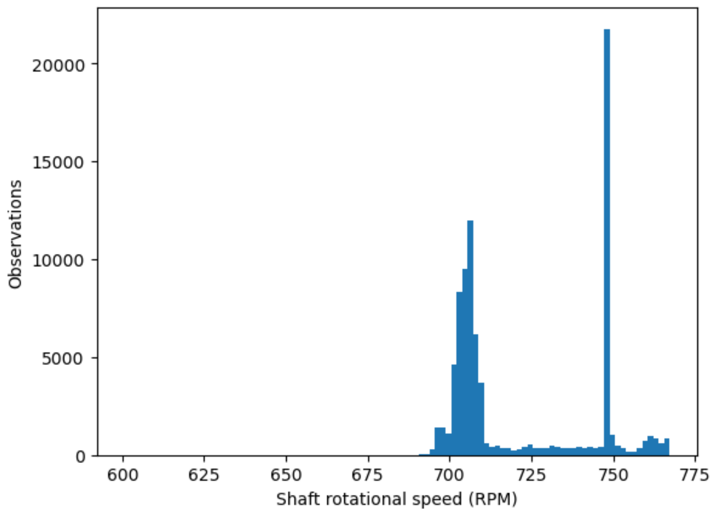


Figure 5: Statistical distribution of the shaft rotation speed variable.

dropped out entirely. The regulation chamber level is not recorded and is zero for all observations. The water level of the Oberaar Lake is also recorded twice due to safety reasons, so we simply drop one of the variables.
The full list of independent variables we use to predict the behaviour of the machine can be seen in table 2 in the appendix.

3 Non-Temporal Models

Having the two datasets at our disposal, we can exploit correlations between the datasets to build an anomaly detection system. First we can build machine learning models to predict the vibrations in the rotordynamic dataset from the data in the operational dataset. Then differences between the predicted and observed vibrations can be flagged as anomalies. The models predict the vibrations in the machine based on the conditions it's subjected to. Hence, we use the rotordynamic dataset as our dependent variables and the operational dataset as our independent variables since we want to model the way external factors (temperatures, water levels in pipes, active power of turbine/pump, etc.) impact the observed vibrations of the shaft.
We train 3 such models each increasing in complexity. The first is a simple linear regression model which only predicts one dependent variable at a time. Next is a non-linear neural network capable of predicting multiple variables. Finally, we create a more complex neural network capable of predicting the complete motion of the shaft in 3d space along with the RMS of the vibrations in the bearing housings.

3.1 Linear Model

We decide to first construct a simple linear regression model to investigate the linear correlations between our dependent and independent variables. One of the key benefits of linear regression models is explainability. However, we have an issue of high dimensionality. If we generate a model with all our data, each one of our 189 dependent variables

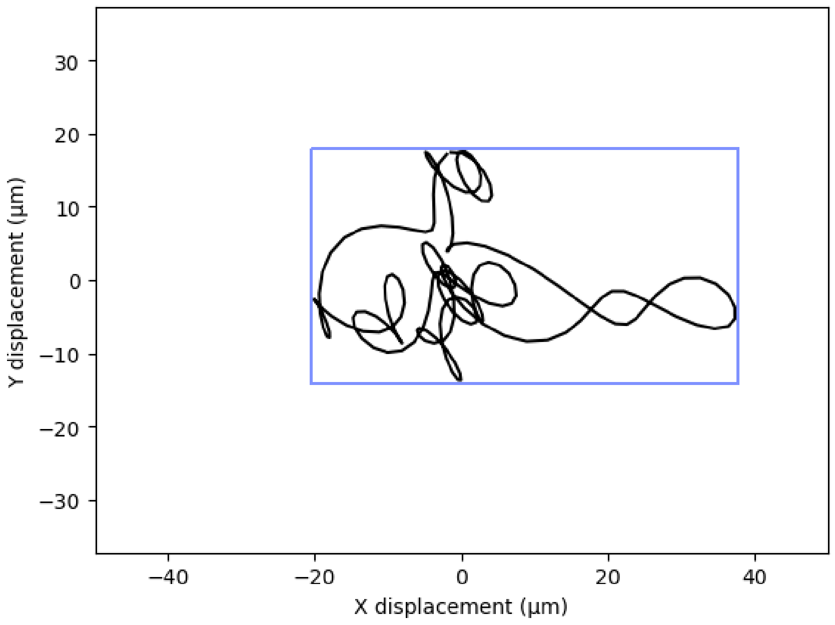


Figure 6: An orbit plot with an enclosing rectangle plotted around it.

will have 61 coefficients each. Additionally, each of these 189 variables only contributes to a part of the total vibrations in the machine. Moreover, if we ran into problems while attempting to predict all of these, we would probably struggle to resolve them due to difficulties in interpreting the model. Therefore we decide to start small and reduce the dependent variables to a single one. This is the typical use case for linear regression and our Python library (statsmodels) made it easy to work with linear regression on one single variable. This helps us quickly train and evaluate preliminary models to explore the relationships between our two datasets.

Variables

Since the goal is ultimately to predict the complete motion of the shaft, we decide to instead predict a single variable that is somehow representative of that motion. However, choosing a single variable is hard because of the complex shape of the vibrations. Additionally, due to the directional information in the 2d Fourier data, vibrations of similar severity can have quite different orbit parameters. Although orbits are measured in 3 dimensions — laterally in two dimensions and axial in the third — the lateral dimensions are likely more significant and therefore we decide to drop the axial dimension. We decide to characterise the vibration by calculating the perimeter of the rectangle surrounding the orbit. This is easy to compute by summing the peak-to-peak values for both dimensions. Greater vibrations produce greater perimeters and so this is a quick proxy for the severity of the vibrations in the machine. This is visually depicted in figure 6. We use similar methods to generate other variables for the bearing housing. We also try summing both dimensions of the RMS data or both dimensions of the first frequency window (1x harmonic) of the Fourier data to generate variables.

Results

Table 1: OLS Regression Results

Dep. Variable	Total_Amplitude_Pump
R-squared	0.722
Model	OLS
Method	Least Squares
No. Observations	610840
Df Residuals	610778
Df Model	61
F-statistic	2.595e+04
Covariance Type	nonrobust

As each model only predicts a single variable, we have to create a new model for each, resulting in many models to predict vibrations at different combinations of pump or turbine and bearing or shaft. This is further multiplied by choosing which of the aforementioned variables to predict.
For brevity, we cover here only the results from predicting the summation of the two dimensions of the 1x harmonic from the Fourier data. It's abbreviated in the model results to 'Total_Amplitude' and we'll refer to it as such here. We can see in table 1 that the R-squared value of the total amplitude of the pump is 72.2%. Thus, the regression model explains approximately 72.2% of the variance in the predicted variable. The F-statistic showing the overall significance of the model is 2.595e+04 ($p < 0.01$), indicating that there is definitely a correlation between the operational dataset and the total amplitude. The model produces good results for starters, however when we change the dependent variable to be the total amplitude computed for the turbine the results drastically worsen. Running the same regression model on the total amplitude of the turbine instead of the pump results in a R-squared of 42.5%. Hence, only 42.5% of the variability in the amplitude of the turbine at the first harmonic is explained by the operational data. The F-statistic is also notably worse at $7.386e{+}03$. The results for predicting the sum of both dimensions of the RMS do not differ much.
While the results do show that correlations between the two datasets clearly exist, one will expect a higher R-squared value for such a tightly coupled mechanical system. Reasons for under-performance could be that there are non-linear relationships in the data or that our data preparation is not thorough enough for such a simple model. In any case, a single dependent variable does not give much insight so we decide to move on to a more capable model.

3.2 Simple Non-Linear Model

To exploit non-linear models, we move on to neural networks. We use the same independent dataset from the merged and filtered operational data. The benefit of this sort of model is that it is able to learn non-linear relationships in the data without much human

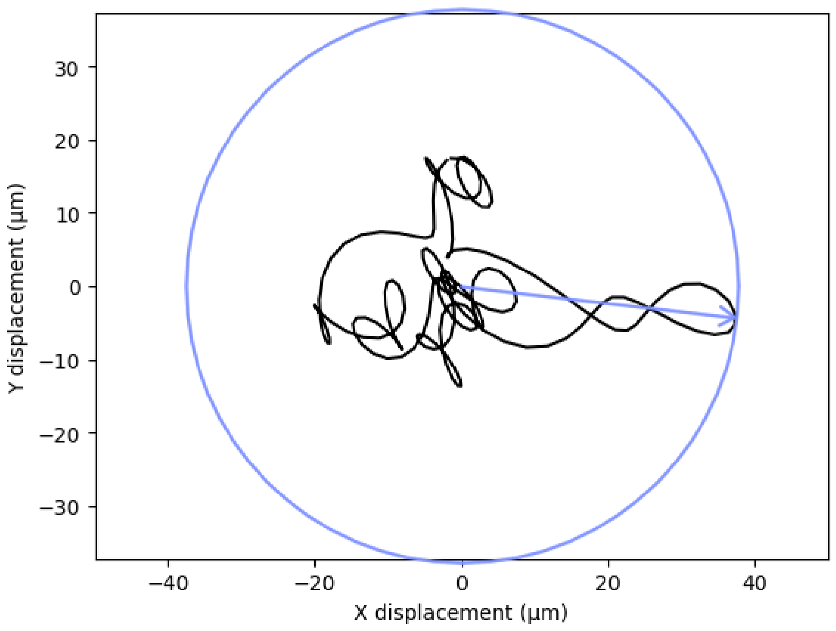


Figure 7: Maximum displacement of the orbit measured from the center point.

input. One example of this pertaining to our data is the power consumed/produced independent variable. This is represented by one variable which is positive when producing power and negative when consuming. These two ranges represent totally different states of the machine as power production occurs when the turbine is active, and consumption occurs when the pump is active. A neural network is able to observe the value of this variable and model the system totally differently depending on whether it is positive or negative. This sort of modeling is not possible with a simple linear regression without manually splitting data and creating two different models.

Variables

We want to find a more accurate variable describing the vibrational severity. The earlier variables suffer from a directional bias as they simply sum the two dimensions. We can see in figure 6 that rotating the orbit by 45 degrees will produce a much larger perimeter value even though the scale of the orbit is unchanged. A good candidate for a representative variable will be the maximum displacement seen along the entire course of the orbit. This can be thought of as the radius of the circle which encloses the full orbit as depicted in figure 7. We refer to it as the maximum displacement.
To calculate the maximum displacement, we must reconstruct the orbit from the values provided to us in the frequency domain from the output of the Fourier transform. We start with the phases and amplitudes of the different harmonics. The amplitudes obtained from the Fourier transform represent the magnitudes of each harmonic component, indicating the strength or intensity of the vibration at a specific frequency. The phases obtained from the Fourier transform indicate the relative positions or angles of each harmonic component within the signal, determining their alignment relative to the start of the shaft's rotation as determined by the keyphasor.
To generate an orbit we start at the angle $\phi = 0$ and sample all of the 7 different harmonic

signals with the phases and amplitudes specified in the data, summing them together to get the final displacement at the start of the period. We then increment ϕ and repeat the process, iterating until we have covered a full rotation. This is done for both dimensions, and the end result is a sequence of 2d vectors which represent the position of the shaft over the course of one full rotation.
The phase interval between each successive iteration is important as it determines the total number of vectors calculated in the sequence and thereby determines the resolution of the generated orbit plot. We decide to choose a value of $\Delta\phi = 0.03$ radians which provides us with 210 vectors per orbit plot. This is chosen since it is the value where it becomes hard for us to see straight segments on the generated orbit plots, so we reason the resolution is acceptable.
Mathematically we are calculating:

$$v_1, v_2 \cdots v_{209}, v_{210} \quad \text{where} \quad v_n = \begin{bmatrix} v_{n,x} \\ v_{n,y} \end{bmatrix} \in \mathbb{R}^2$$

In this case, x and y refer to the two different dimensions we are dealing with. In the case of bearing housing measurements, the two dimensions are actually named x and y.
We have 7 harmonic frequencies: $H = \{1, 2, 3, 4, 5, 7, 17\}$. Let $i \in \{1, ..., 7\}$ and let $a_{i,x}$ refer to the 7 amplitudes of the harmonics in the x measurements and so on for $a_{i,y}$. Let $p_{i,x}$ and $p_{i,y}$ refer to phases for each of the harmonics. Let $f_i \in H$ refer to the order of the harmonic. We also have:

$$\phi_1, \phi_2 \cdots \phi_{209}, \phi_{210} \quad \text{where} \quad \phi_n = (n-1) \times 0.03$$

Now we can calculate each v_n as follows:

$$v_{n,x} = \sum_{i=1}^{7} a_{i,x} \cdot \cos(\phi_n \cdot f_i + p_{i,x}),$$
$$v_{n,y} = \sum_{i=1}^{7} a_{i,y} \cdot \cos(\phi_n \cdot f_i + p_{i,y})$$

Using this formula, we can build plots of the motion of the shaft or bearing housing at the turbine or the pump. The axial motion is one-dimensional, so we can plot it as well with a one-dimensional version of the same algorithm. Accounting for all these cases, we can now generate five new dependent variables representing the maximum displacement detected at each of these five locations. A table of all the dependent variables can be seen at table 3 in the appendix.

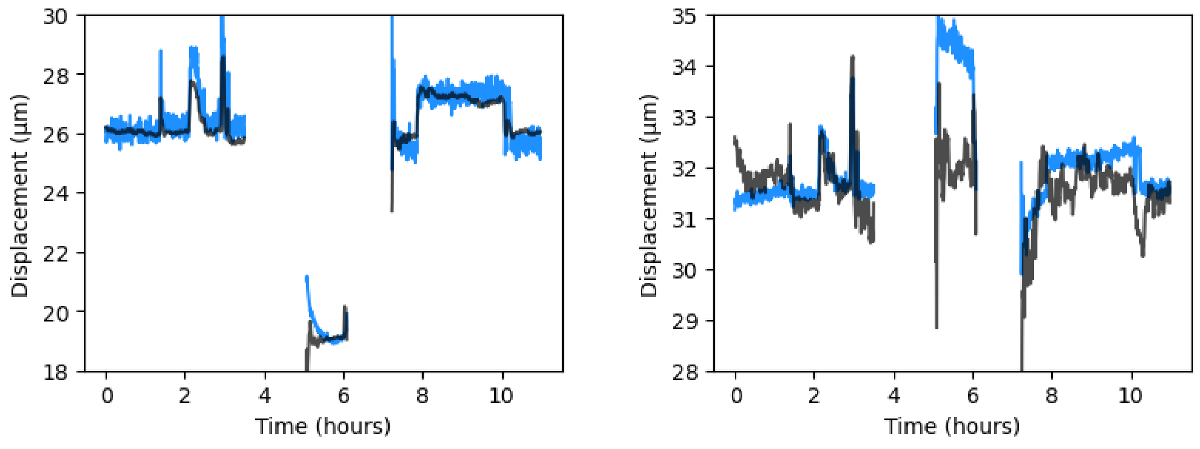
Architecture

In our study, we employ the merged filtered operational dataset as our independent variables, encompassing a total of 61 variables (corresponding to the number of input neurons in our model). Our objective is to predict five specific variables related to the maximum displacements for the shaft at the pump and turbine, the bearing housing at the pump and turbine, and the axial displacement of the shaft.

To accomplish this, we utilise a neural network architecture that consists of one hidden layer comprising 64 neurons. These neurons are fully connected to the 61 input neurons and are also connected to the 5 output neurons responsible for predicting the vibration displacements. The rectified linear unit (ReLU) activation function is used as a non-linear activation function, enabling the model to capture non-linear patterns and variations in the data.
To evaluate the performance of this model, we employ the mean squared error (MSE) as a loss function. The MSE provides a measure of the average squared difference between the predicted and actual values, enabling us to assess the accuracy of our predictions. We split the data into train, validation and test subsets (60%, 20%, 20%). This allows us to be sure that the model is not overfitting the training set. Additionally, we select the Adam optimiser, known for its effectiveness in handling large datasets and optimising networks without the need for much hyper-parameter tuning. We use mini-batches with a batch size of 32 to update multiple times per epoch.

Results

During the training process, we observe a consistent decrease in the validation loss per mini-batch as the number of epochs increases. Validation loss is computed with data that the model has not trained on, so this indicates that our model is not overfitting the training data. Once the training is completed, we go a step further and visually examine the values obtained. By visualising the results, we gain a deeper understanding of the impact of our training efforts, allowing us to gain insights into the patterns and relationships learned by the model. This visualisation aids in the evaluation of the model's performance and assists in identifying any potential areas for further refinement or exploration. We can see in figure 8 how our predicted and measured (true) values change over a period of time. Our predicted maximum pump displacement is fairly accurately placed within the range of observed values and follows the measured values also as spikes occur. Predicting the maximum displacement for the other four variables (at shaft and bearing housings at the turbine and and pump) produced similar results. However, the predictions for the turbine shaft are noticeably worse, wandering more and sometimes even predicting values fully outside the range of observed values (see figure 8b). Even so, this model is a significant improvement from the linear model in almost every aspect.
So far, we achieve very good results in predicting the displacement of the shaft utilising all harmonic information measured in the rotordynamic dataset. However, this maximum displacement metric that we use for dependent variables does not train our model to predict the direction in which the displacement occurs. Since it takes only the magnitude of the furthest point from the centroid it does not take into account at what angle this displacement has happened. The direction of the displacement can be quite useful in detecting anomalies since an anomaly can cause vibrations out of phase with the typical vibrations. Furthermore, the frequency domain is not predicted at all. If, for example, an anomaly occurs where we see large oscillations in the 17th harmonic at a moment in time where we will instead expect to see large oscillations in the 1st harmonic, this model has no way of detecting that. It is also not possible to reconstruct the original orbit from the predicted variables, which is useful for visualisation. As a next step we move to a more complex model which solves all of the issues of our simple non-linear model, by comparing



(a) Pump displacement (b) Turbine displacement

Figure 8: Measured (blue) vs. predicted (black) variables.

complete orbits instead of only using displacements.

3.3 Complex Non-Linear Model

Next, we decide to train a larger model which is capable of addressing the deficiencies in the previous model. It's important to predict many more dependent variables to enable a more comprehensive analysis the orbits rather than focusing solely on maximum displacements. By considering the entire orbit, we gain the ability to use directional and frequency information for anomaly detection purposes.

Variables

To predict the complete motion of an orbit, we need to predict all the Fourier information stored in the rotordynamic data set. This expanded set of variables enables us to reconstruct the complete orbits. This approach requires the network to develop a more holistic 'understanding' of the vibration patterns and provides us with a means to evaluate the accuracy and effectiveness of our predictions by comparing orbits both numerically and visually. The extra information also allows users of the model the potential to diagnose the nature of detected anomalies based on the precise nature of the difference between the predicted and observed values.

Hence, we use as dependent variables all harmonics of the shaft and axial movements. For the bearing housings we take the RMS because the data is far noisier. We verify this by plotting sequential orbits for both the bearing housing data and the shaft data. Sequential shaft orbits look similar and show a progression through time, while each sequential bearing housing orbit looks essentially random, perhaps with a similar size. One likely explanation for this is due to the fact that the data is captured by accelerometers and integrated to get position which is inherently noisy and less accurate than direct position measurements.

Each orbit consists of 28 variables, being 7 harmonic frequencies $\times$ 2 dimensions $\times$ 2 values (amplitude and phase). Axial motion is only in one dimension so it only requires

14 variables. Bearing housing RMS values are 14 variables (7 harmonic frequencies $\times$ 2 dimensions). In total, we now have 98 dependent variables instead of 5. A table of all the dependent variables can be seen at table 4 in the appendix.

Architecture

We train a fully connected neural network but this time with two hidden layers each with 128 neurons. The non-linear function is again ReLU and the loss function and optimiser are again the MSE and Adam.

Further Data Preparation

The independent data is again the filtered merged operational dataset of 61 variables, but now the output layer is the dataset of the 98 chosen dependent variables describing the orbits. Once again, we split our dataset to train, validation and test subsets (60%, 20%, 20%) and evaluate our training and validation losses.
However, we also use our previous work to enhance our data filtering. We once again calculate the maximum displacement metrics for our data and use them to filter out bad data. Manual inspection of the data shows that there are some data points with nonsensical Fourier parameters that lead to very large orbits. For example, while a typical orbit is around 40 microns across, there are single data points which produce orbits over 20,000 microns across. This is orders of magnitude larger than what is physically possible. These are likely the result of errors in the data acquisition process, so we filter out all data points which have orbits that are larger than the 99.9th percentile of orbits. We also filter out data with orbits of zero displacement.
Further, the Fourier data is stored using phase and amplitude (polar coordinates). This presents a problem to the model since phase measurements are not continuous. This is an issue with all polar coordinate systems since 360 degrees must wrap around to 0 degrees. This discontinuity breaks training as the variable is not fully differentiable. As a result, we make sure to convert this data to the Cartesian coordinate system.

Results

The training loss is evaluated on every mini-batch for every epoch, thus showing more variance. The validation loss is evaluated once per epoch on the entire validation set. We use 100 epochs for training and with more epochs both training and validation loss get lower. However, after epoch 20 they start to go down slower but the validation loss continuously decreases indicating that we are not yet overfitting.
A current problem that remains is the distribution of pump and turbine activity. In our two previous models our predictions for the vibrations of the shaft at the pump are considerably better than the ones at the turbine. This is potentially due to the fact that we have more pump data than turbine data, as seen in figure 9. Across the entire dataset, there are roughly 3 times more data where the pump is operating as opposed to the turbine.

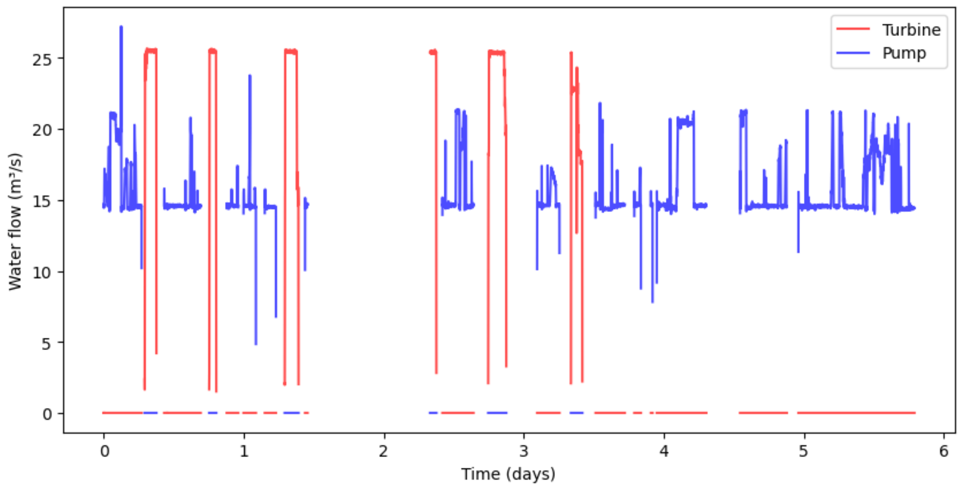
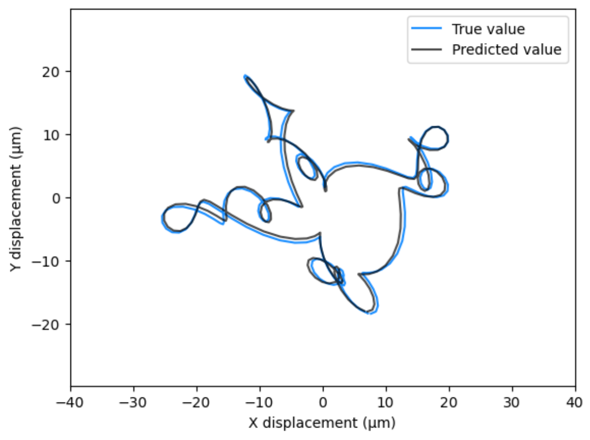
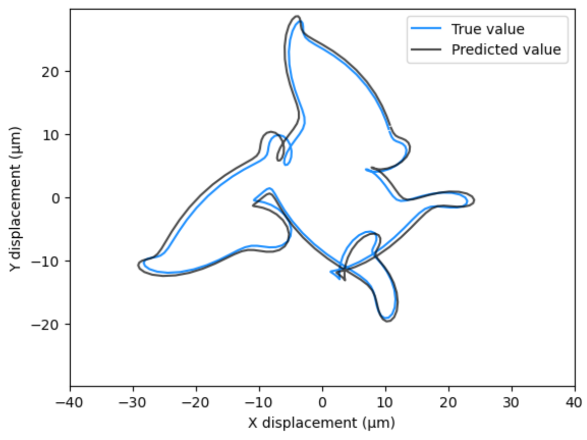


Figure 9: Water flows through the pump and turbine. Dates have been anonymised.

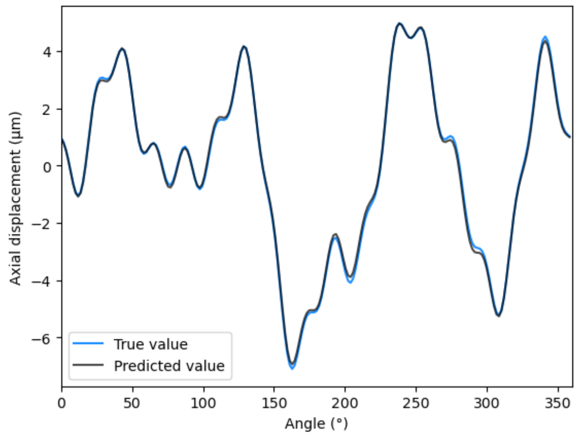
Thus, as a next step, we train two models which are trained only on either pump data or turbine data. However, we find this doesn't make any appreciable increase in performance. Additionally, we also try different independent data subsets from the operational dataset. We try using first only the direct operational data and then adding different combinations of the water and temperature datasets. An interesting but also expected behavior that we see is that temperatures have the largest effect on improving predictive power whereas water levels produce only a slight increase in performance. We find that adding water levels also introduced a slight overfitting tendency for the turbine predictions. We can guess this is because the model is already struggling to predict the turbine with the available data, and the water levels provide an extra degree of freedom to memorise the measured values.



(a) Pump orbit



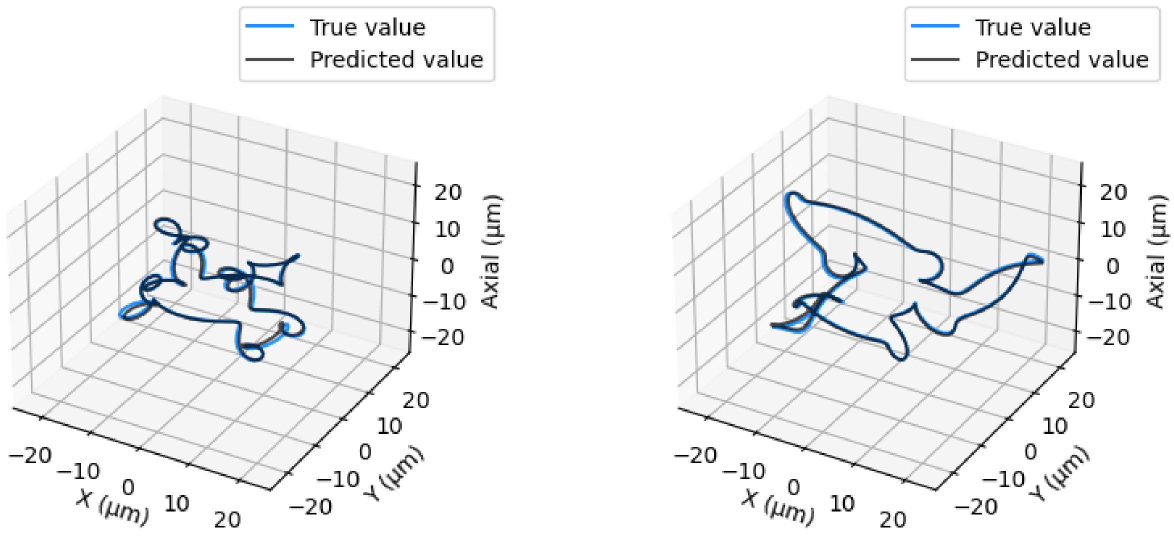
(b) Turbine orbit



(c) Axial motion of the shaft

Figure 10: Comparison of pump and turbine vibrations.

This complex model enables us to also evaluate the errors visually to see how our predicted values differ from the true values by comparing the orbits. We show the predicted orbits of the shaft for both the pump and the turbine for a randomly chosen time-step. We can see in figure 10 that both of the orbits have quite complex shapes. We can also see that the two orbits differ a lot from each other. The turbine orbit (fig. 10b) seems to be more smooth than the pump vibration orbit (fig. 10a). Most importantly, our predicted orbits are quite accurate.
While the orbits allow us to observe and analyse the radial movements of the shaft, the axial displacement remains unseen. However, it is an essential aspect to consider as it represents the lengthwise motion of the shaft along its axis. We can see in figure 10c that our model also predicts this variable with reasonable accuracy.
Combining the orbit with the axial deviation results in a sequence of three-dimensional vectors. A 3d orbit plot can be seen in figure 11 for both a turbine and a pump data point. In the plots the X and Y directions are the 45° and 135° sensor data which we use to construct the orbit and the Z direction is the axial displacement. These plots are generated at the same time-step as the 2d plots, and so we can see the similar shapes. Anomaly detection can be accomplished by comparing the difference between the predicted and observed data via a metric such as the MSE and flagging anomalies if the difference



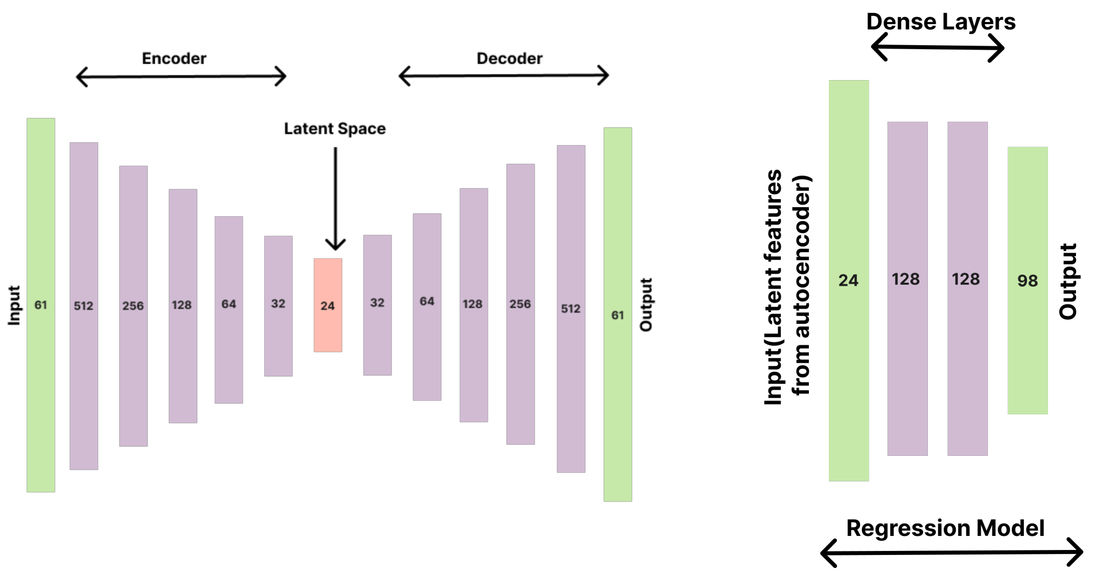
(a) Pump vibration fingerprint (b) Turbine vibration fingerprint

Figure 11: 3D plots of vibrations at the pump and turbine.

exceeds a certain threshold.

4 Autoencoder based Regression Model

We also propose an innovative approach utilising an autoencoder-based regression model[4] for vibration analysis and anomaly detection. By leveraging the capabilities of autoencoders, we can extract meaningful features, reconstruct signals and generate predicted orbits. We build this on top of the previous complex neural network model.
The primary advantage of utilising an autoencoder is its ability to reduce the dimensionality of the input data. We achieve a more concise and meaningful feature space by training the autoencoder to compress 61 input features into a lower-dimensional representation of 24 features. This latent space captures the most salient features and helps us reduce the dimension of the input data. Furthermore, autoencoders excel at capturing non-linear relationships within the data, which is crucial for accurately modelling complex vibration patterns. Another significant benefit of this approach is its robustness to noisy or incomplete data. During training, the autoencoder learns to encode and decode the input features, effectively handling missing or corrupted values. The robustness enhances the overall accuracy and effectiveness of the model.
Autoencoders can also be utilised as anomaly detectors. An autoencoder, trained on normal or healthy data, aims to reconstruct the same input. When presented with new data, the autoencoder attempts to reconstruct it based on the learned normal patterns. If the reconstruction error is high, leading to a deviation from this learned pattern, it can be flagged as an anomaly.



(a) Autoencoder architecture (b) Regression Model architecture

Figure 12: Autoencoder used as a feature extractor.

4.1 Architecture

Our autoencoder architecture depicted in figure 12a consists of multiple dense layers with varying numbers of neurons. The input layer receives data with 61 input features. Following the input layer, we have a dense layer with 512 neurons, connected to a layer with 256 neurons, and subsequently to a layer with 128 neurons. The network continues with layers of 64, 32, and 24 neurons, respectively. The next layers include a dense layer with 32 neurons, followed by layers with 64, 128, 256, and 512 neurons. Finally, the output layer has 61 neurons, matching the dimensionality of the input data. This architecture allows the autoencoder to learn a compressed representation of the input data in the bottleneck layer with 24 neurons and then reconstruct the original data with minimal loss in the subsequent layers. After experimenting with various dimensions of the bottleneck layer, we find that a bottleneck layer size of 24 neurons yields the best results in capturing and reconstructing the essential features of the input data. This model is trained over 200 epochs.

4.2 Regression Model

Having trained the autoencoder on the independent data, we strip the decoder part and use the trained encoder as a feature extractor to extract lower dimensional meaningful features from the input data. Using these extracted latent features, we proceed to train our complex model. The complex model is the same as the one defined in section 3.3 and trained in the exact same way but with the extracted features. The model architecture can be seen in figure 12b.
The performance of the current model, which utilises the features extracted from the autoencoder, exhibits slight improvement compared to the previous complex model. However, it does not demonstrate a significant performance boost. The results are similar to

the one obtained from the complex model itself. The limited performance boost may be attributed to the inherent limitations of the autoencoder itself. The compressed representation obtained from the autoencoder may lose some fine-grained details that can be vital for achieving substantial performance improvement.

5 Recurrent Neural Network

In the next step, the time dependency of the data is taken into consideration by using a recurrent neural network. Here the dataset is prepared as described in section 2.3. Moreover, this chapter compares a recurrent neural network (RNN) and a fully connected neural network (FCNN) with respect to their accuracy.

5.1 Background RNN

A RNN[3] is a type of neural network designed to process sequential data. It has a unique ability to capture and utilise information from previous steps, making it ideal for tasks involving sequences, such as natural language processing and time series analysis.
An RNN consists of a chain-like structure of nodes called cells. Each cell takes an input, produces an output, and passes information to the next cell. The key feature of an RNN is its hidden state, which serves as its memory. The hidden state is updated at each step and contains information about the previous inputs in the sequence. During the forward pass, the RNN processes the input sequence one element at a time, updating its hidden state at each step. This allows the network to retain information from the past and use it to understand the current input. The recurrent nature of the network enables it to capture both short-term dependencies within a local context and long-term dependencies spanning the entire sequence.
A schematic of a RNN is depicted in 13. It shows how a RNN is constructed internally and that the hidden state is continuously updated by the variable C.
RNNs offer several advantages when it comes to time-dependent data. One key advantage is their ability to capture and utilise temporal dependencies in the data. RNNs can retain information about past inputs through their hidden state, enabling them to make predictions or generate outputs based on the context of previous time steps. This makes RNNs well-suited for time series forecasting, speech recognition, and language modelling tasks. Additionally, RNNs can handle variable-length input sequences, allowing them to process data of different lengths without needing fixed-size inputs. This flexibility makes RNNs powerful tools for analysing and modelling time-dependent data.
Due to the abovementioned advantages, RNNs might also be a good fit for anomaly detection in our hydroelectric power plant. Therefore, the following describes training an RNN and comparing it to a similar FCNN.

[1]`https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn`

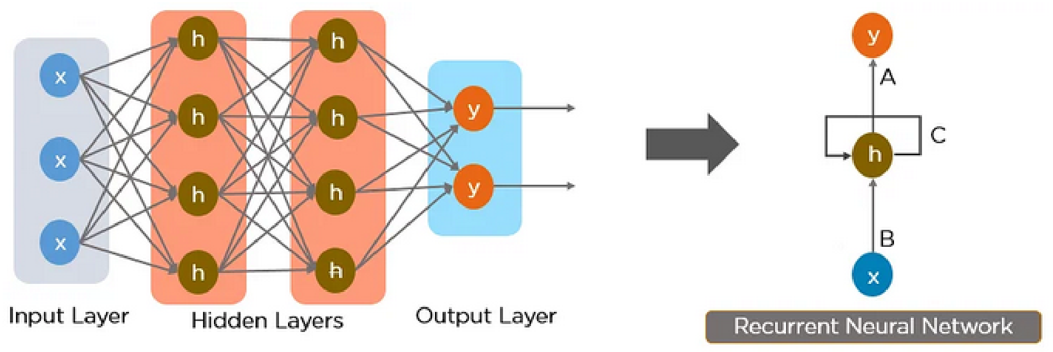


Figure 13: Schematic structure of a RNN adapted from Avijeet Biswal's tutorial on RNNs[1].

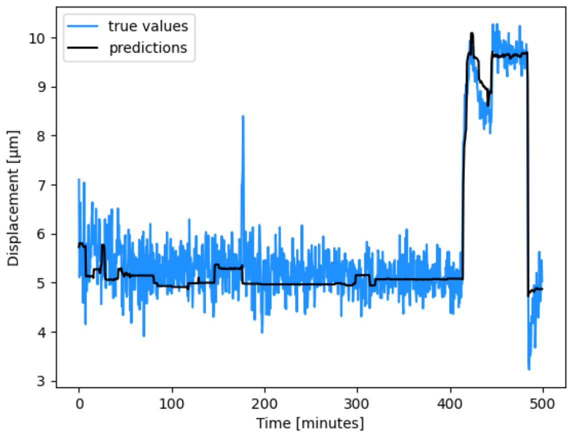
5.2 Comparison between a RNN and a FCNN with a Sliding Window

An FCNN that uses a sliding window approach processes sequential data using a fixed-size window to extract features from the input sequence. The window slides across the sequence, extracting a subset of data at each position. These extracted subsets are then fed into an FCNN for processing. Thus, similar to an RNN, it takes a series of data as input. However, this approach cannot explicitly model the temporal dependencies within the sequence. Each window operates independently, without considering the context of past or future data points outside the window. As a result, the model may struggle to capture long-term dependencies and make accurate predictions. On the other hand, an RNN is specifically designed to handle sequential data by retaining memory of past inputs through its hidden state. The hidden state is updated at each time step, incorporating information from the current input and the previous hidden state. This enables the RNN to capture and utilise the temporal dependencies present in the sequence. The RNN can process variable length inputs and maintain a dynamic internal representation of the sequence. This makes it better suited for tasks that require an understanding of long-term dependencies, such as time-dependent data coming from the operational and rotordynamic datasets.
In this comparison, the MSE loss is opted as the loss function and Adaptive Moment Estimation (ADAM) as Optimiser. For better comparability, as many parameters as possible are chosen to be equal, such as input size, hidden size, output size, batch size, etc. The dataset is split to train, validation, and test subsets with a respective percentage of 80%, 10%, and 10%.

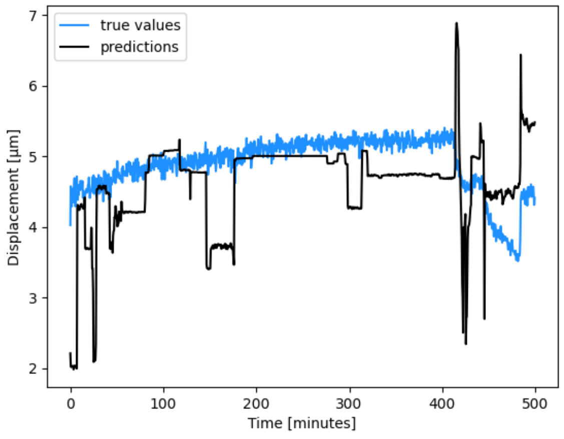
Performance of the RNN

Similarly to the complex non-linear model, one hundred epochs are again used for training. The training loss, as well as the validation loss, are decreasing while the number of epochs is increasing. Also, the test loss is low, similar to the final validation loss. The training error is evaluated on every mini-batch for every epoch, whereas the validation loss is evaluated per epoch.

We can see in figure 14a the true vs predicted displacement of the shaft during pumping mode. The blue line represents the true values, incorporating a certain amount of noise. The black line represents the prediction. It is clear to see, that the blackline describes the blue line quite well. Also, when there are spiky outliers in the true values, e.g., when there is a gap in the dataset, the black line covers that.



(a) Truth vs prediction of displacement of the shaft during pump mode in RNN.

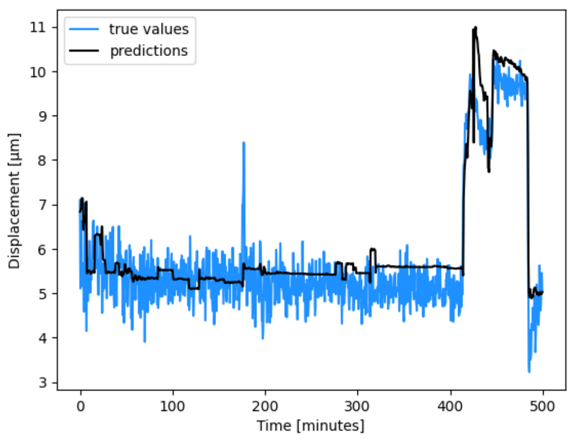


(b) Truth vs prediction of displacement of the shaft during turbine mode in RNN.

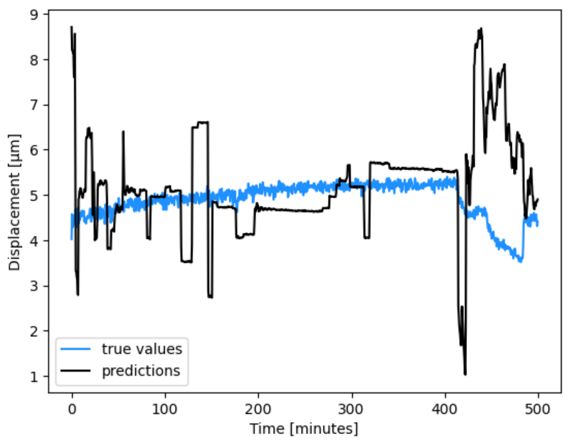
The prediction of the shaft displacement is worse in the turbine mode, as can be seen in 14b, due to less data for the turbine mode that we have at our disposal. There, the black line suggests outliers where there are none in reality. All in all, the course of the blue data is not as good covered as in pumping mode.

Performance of the FCNN with a Sliding Window

Analogously to the RNN, during the training of the FCNN, both train loss and validation loss are decreasing equally, so the network is not overfitting. Equally to the RNN, the training error is evaluated on every mini-batch for every epoch, whereas the validation loss is evaluated per epoch.
Various numbers were tested for size for the sliding window, but 10 seemed most appropriate. This means that 10 consequent data points are simultaneously fed into the NN. In the next time step, the window 'slides' one step further, again covering 10 data points. Equally to the RNN, the same predictions are also made with the FCNN. Both figures 15a and 15b show that predictions are more precise in pump mode than they are in turbine mode.



(a) Truth vs prediction of displacement of the shaft during pump mode in FCNN.



(b) Truth vs prediction of displacement of the shaft during turbine mode in FCNN.

Conclusion of comparison

Even though the training of the RNN is spikier than the training of the FCNN, the RNN reached a lower loss. Both models perform well in predicting the displacement of the shaft during pumping mode but worse during turbine mode. As mentioned before, this is because there is three times more pump data available than turbine data. The training curves are smoother than the ones in the previous chapters because the batch size is significantly larger this time. Hence, any peaks are smoothed out.

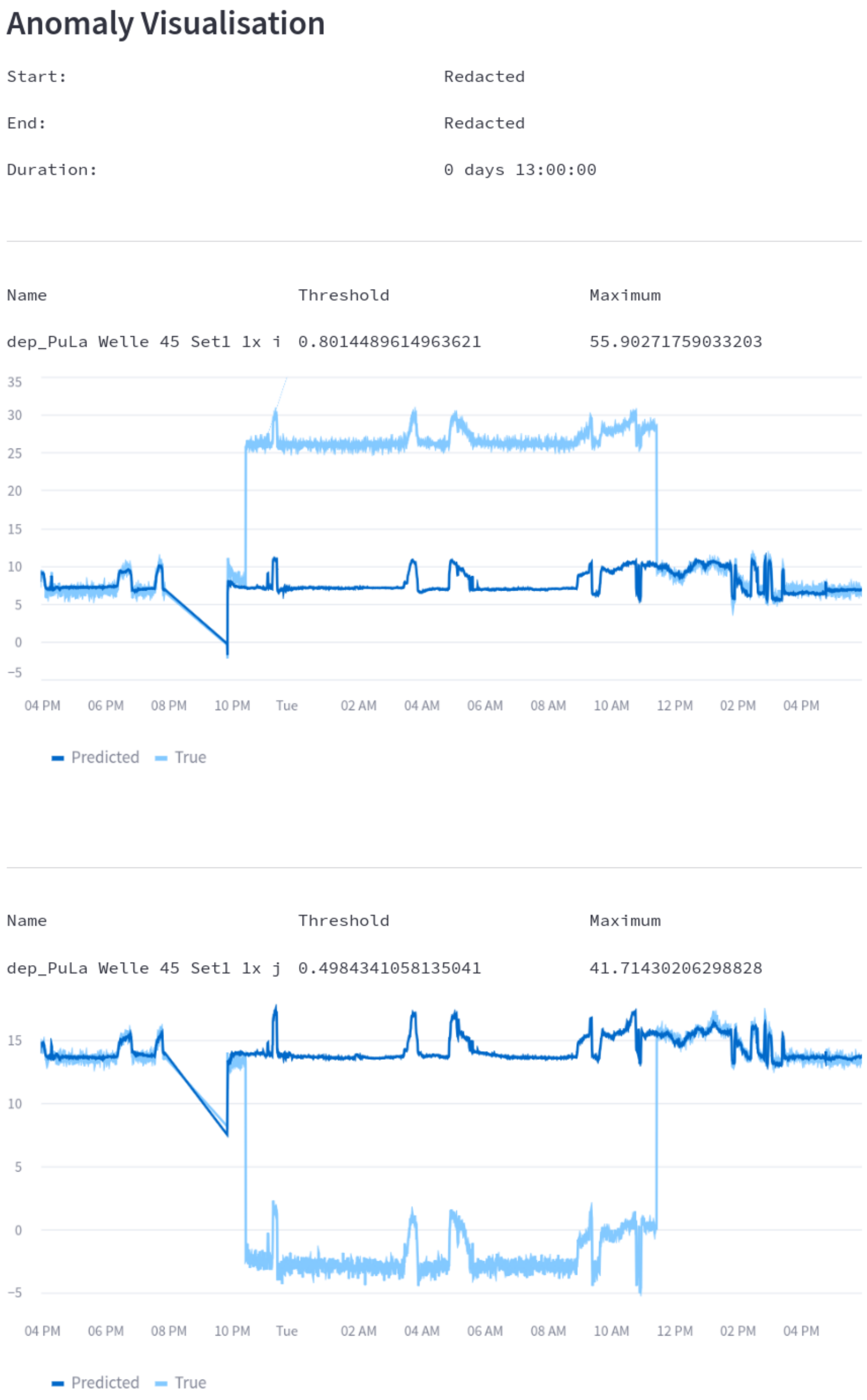
6 Anomaly Detection

We built a simple anomaly detection system capable of detecting anomalies using the output predictions of any of the models. IfTA provided us with a new, manipulated dataset that allowed us to test our trained models on unseen data. We tested our detection system against this dataset with hand-crafted anomalies.
The system uses a simple threshold-based technique combined with filtering to flag variables and time spans that seem anomalous. The variable is considered anomalous when the error between the predicted and measured variables exceeds the threshold.
First, the thresholds must be set. Since each variable has a different variance and prediction quality, we must generate a unique threshold for every variable. This is accomplished by calculating the error for each data point in the entire training dataset and then taking a specific quantile. In our case, we choose the 0.985th quantile, meaning that only errors larger than 98.5% of all errors seen in training are considered anomalous. This quantile must be selected carefully; if it is too large, anomalies will go undetected, and if it is too small, regularly occurring differences between the predictions and observed values will be marked as anomalies. Therefore this is also affected by the quality of the predictions. The better the model, the smaller the threshold can be, and smaller anomalies can then be detected.
Secondly, for each variable, each anomalous time-span is filtered for length. Short sequences exceeding the threshold are common and can be due to many factors such as machine operation starting or stopping or erroneous readings from the sensors. There-

fore, we mandate that each sequence of data points exceeding the threshold must be of a certain length to be considered as an anomaly. In our case we choose 30 samples equating to 15 minutes in our model. In future systems, this can be scaled depending on the size of the error, so larger anomalies are detected earlier.
Lastly, different variables that have anomalies at the same time are grouped together and a report is generated showing the start time, end time, variables, thresholds, and observed maximum errors for each anomaly. This report is saved to a file and can be read by a web application built with the Streamlit framework which allows the user to step through and see plots of each anomaly. A screenshot of this application depicting an artificial anomaly detected by our models can be seen in figure 16.
The anomalies contained in the new, manipulated dataset have been detected by the more sophisticated approaches presented in this report (complex non-linear model, autoencoder and RNN). Depending on the threshold, the anomaly detection system has found a different number of anomalies. If the threshold is comparatively low, the system flags many sequences, even the ones that have not been manipulated but have a slightly larger gap between prediction and true value; if the threshold is too high, the system might not detect anomalies.
In total, the manipulated dataset contains three artificial anomalies. The detection system is able to successfully flag two of these three anomalies in the dependent variables. The third anomaly is in an independent variable — one of the water pressure readings for the turbine is artificially set to zero. This is problematic for our model as our turbine predictions are relatively low quality, and therefore thresholds for turbine variables are quite high. Furthermore, the turbine is not in use during the majority of our dataset, therefore it is quite typical for this variable to be zero. These factors likely contribute to our system failing to detect this anomaly. This could potentially be improved with more turbine data. Failing that, more advanced methods such as predicting the distribution of turbine data rather than absolute values could be a promising path.

7 Conclusions

In this report, we explore various methods to automate the diagnosis of mechanical issues in the machinery of a pumped storage hydroelectric plant. Throughout the project, we employ various machine learning models, including linear regression, non-temporal neural networks, RNNs, and auto-encoders, to train and predict machine vibrations from the given operational data.
After training and evaluating our models, we perform a blind test on an altered dataset provided by IfTA. The test delivered promising results, detecting two out of three anomalies. Though this is already an achievement, the failure to detect one of the anomalies highlights room for further development. The improvement of turbine predictions could be explored through the use of more advanced models and more varied data. The use of labelled data and better data preparation techniques could also provide improved results all around through superior predictive models and lower thresholds.
In conclusion, the project represents an example of how machine learning and data-driven techniques can optimise the operation and maintenance of vital energy infrastructure.



Anomaly Visualisation

Start:	Redacted
End:	Redacted
Duration:	0 days 13:00:00

Name	Threshold	Maximum
dep_PuLa Welle 45 Set1 1x i	0.8014489614963621	55.90271759033203

Name	Threshold	Maximum
dep_PuLa Welle 45 Set1 1x j	0.4984341058135041	41.71430206298828

Figure 16: A screenshot of our Streamlit application presenting a detected anomaly.

References

[1] California Independent System Operator (CAISO). *Understanding the Duck Chart: Managing a steep ramp with net load.* 2013.

[2] Paul Denholm and Erik Ela. *The Role of Energy Storage with Renewable Electricity Generation.* Tech. rep. National Renewable Energy Laboratory (NREL), 2010.

[3] Jeffrey L. Elman. “Finding Structure in Time”. In: *Cognitive Science* 14.2 (1990), pp. 179–211.

[4] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. “Reducing the Dimensionality of Data with Neural Networks”. In: *Science* 313.5786 (2006), pp. 504–507.

[5] U.S. Department of Energy. *Pumped Storage Hydropower.* 2019.

Appendix

Tables

Table 2: Operational dataset independent variables

Variable	Count
Power produced or consumed	1
Reactive power	1
Shaft RPM	1
Turbine and pump manifold pressures	4
Turbine and pump flow rates	2
Turbine guide vane control position	1
Power produced or consumed by other machines at the facility	3
Bearing temperatures	16
Motor-Generator stator and coil temperatures	12
Seal temperatures	6
Air temperatures	4
Oil temperatures	4
Water temperatures	2
Penstock water level	1
Surge tank water level	1
Reservoir water levels	2
Total	61

Table 3: Simple neural-network dependent variables

Variable	Count
Maximum shaft displacement at the pump	1
Maximum shaft displacement at the turbine	1
Maximum bearing housing displacement at the pump	1
Maximum bearing housing displacement at the turbine	1
Maximum axial displacement of the shaft	1
Total	5

Table 4: Complex neural-network dependent variables

Variable	Count
Shaft orbit Fourier data at the pump	28
Shaft orbit Fourier data at the turbine	28
Shaft axial Fourier data	14
Bearing housing RMS values at the pump	14
Bearing housing RMS values at the turbine	14
Total	98