Predicting Failures in Simulated Turbulence Data with Machine Learning Techniques

Max Seelen

A thesis presented for the degree of
Master of Science

Department of Geosciences
Utrecht University
The Netherlands
Supervisor: Prof. dr. Jeannot Trampert

15th of March, 2024

Abstract

Prediction of earthquakes is a highly complex and difficult task. Traditionally, earthquake prediction uses statistical analysis of historical earthquakes and observation of several parameters such as ground deformation, but none of these methods can provide reliable accurate predictions. Machine learning might help improve the prediction of earthquakes. Machine learning algorithms perform well in predicting lab created earthquakes, but these are highly periodic and therefore relatively easy to predict. These algorithms therefore have to be tested on more complex data as well. The physics of real world earthquakes is poorly understood, and thus simulated data of turbulence where the physics are well understood is used in this thesis. Techniques as Random Forest, Temporal Convolutional Networks and Long Short-Term Memory are tested. Additionally, data analysis is performed to better understand what parts of the data contain the most valuable information for prediction. All algorithms show decent performance in predicting times of failures in the near future, but struggle when the next failure is far in the future. If only medium-long periods of the acceleration data are used, the accuracy of predicting these far in the future failures can dramatically improve, suggesting that only parts of the acceleration data are useful for prediction while the rest can be considered irrelevant. This is also visible in a Hankel Alternative View of Koopman analysis of the data. The artificial nature of the data used makes separation of the periods very easy, which is not the case for real world earthquakes. Principal Component Analysis and Uniform Manifold Approximation and Projection analysis show that the variance of the acceleration also contains useful data for prediction, while the kurtosis of the acceleration is not as useful. With advanced data analysis and filtering techniques, it might be possible to extract useful data from seismograms to predict earthquakes with the help of machine learning techniques.

1 Introduction

The prediction of material failure is a big field of study, both within earth sciences and outside. Within earth sciences, it is of course the prediction of earthquakes that is highly sought after, as well as avalanches or landslides. Outside earth sciences, failure prediction has great value in structural engineering (Sibly, 1977), aviation (Nozhnitzky, 2008), industry (Thangaraj & Wright, 1988) and many other applications. In the past, these predictions were mostly done on the basis of statistical analysis and visual inspection of the material condition. While these methods can give warnings of an incoming failure, allowing people to take appropriate measurements, they usually can not give the exact time of failure. In engineering, seeing warnings of an upcoming failure usually results in the machine or structure taken out of service for repairs or recycling. Of course, this is not an option for earthquakes.

In the case of earthquakes, most forecasting is done with statistical analysis of historical earthquakes (Lomnitz, 1966; Kagan & Knopoff, 1987). These statistical analyses can only give a prediction with a very large margin of error, ranging from days to years or even centuries depending on the size and location of the predicted earthquake. This makes it very hard for governments to take appropriate action for upcoming earthquakes beyond the implementation of building codes and setting up of emergency response plans (Marzocchi & Zechar, 2011). Other methods for estimating the chance of an upcoming earthquake include measurements of ground uplift or subsidence (Suzuki, 1982) and identification of foreshocks (Wang et al., 2005). Neither of these methods allow for high accuracy predictions to be made. Ground uplift or subsidence can be associated with upcoming earthquakes, but not in every case. Many earthquakes also do not show ground uplift or subsidence preceding the event. Foreshocks also do not occur for every earthquake, with only nine percent of moderate to strong earthquakes having direct foreshocks and 51 percent having generalised foreshocks (Wang et al,. 2005). The characteristics of the foreshocks are also highly location dependant and can often be hard to identify as a foreshock until the main earthquake occurs (Lin, 2009).

Earthquake prediction is so difficult because fault systems are usually highly complex with poorly understood physics. It is also usually not possible to do any direct measurements of the stress on a fault, the length of the fault or determine the surrounding material of a fault, and therefore we have to rely almost purely on historical events and seismology. While earthquakes can follow a rough distribution, it is rarely the case that earthquakes have consistent inter-event times. This has lead some researchers to believe that earthquakes are fundamentally unpredictable (Geller et al., 1997).

In recent years however, a different prediction method has risen: machine learning. The idea of using machine learning in earthquake prediction is not new but up until recently, the algorithms have not been able to predict with high accuracy. However, thanks to advancements in computing technology, computers are now able to handle and process more data every year, allowing for the strengths of machine learning to be exploited.

Machine learning specifically for earthquake prediction has a long history. Pattern recognition algorithms were already used in prediction of earthquakes back in the 70's (Gelfand et al., 1972, 1976; Briggs et al., 1977) with limited but promising results. In 1999, Rovithakis & Vallianatos attempted to use a Dynamic Neural Network to identify electric earthquake precursors, again showing promise but not with high accuracy. Since then, prediction with other machine learning algorithms has been attempted, but not with high enough accuracy to be applied in earthquake risk management.

In recent years machine learning has been widely used in the prediction of lab created earthquakes, with high accuracy models developed in 2017 (Rouet-Leduc, 2017). These lab-quakes are created by putting a sample of rock with known dimensions and material properties under high pressures. When the sample then fails, the acoustic emissions are recorded as well as the stress the sample is under. Machine learning techniques applied to these datasets to both predict the time to the next failure and forecasting of stress have been relatively successful in achieving highly accurate results (Laurenti et al., 2022) as well as predicting the magnitude of the failure (Corbi et al., 2019). Figure 1 shows two examples of predictions made with machine learning models. Because these samples are put under highly consistent shear and normal stress, these lab-quakes are usually highly periodical, much more so than real world earthquakes. This

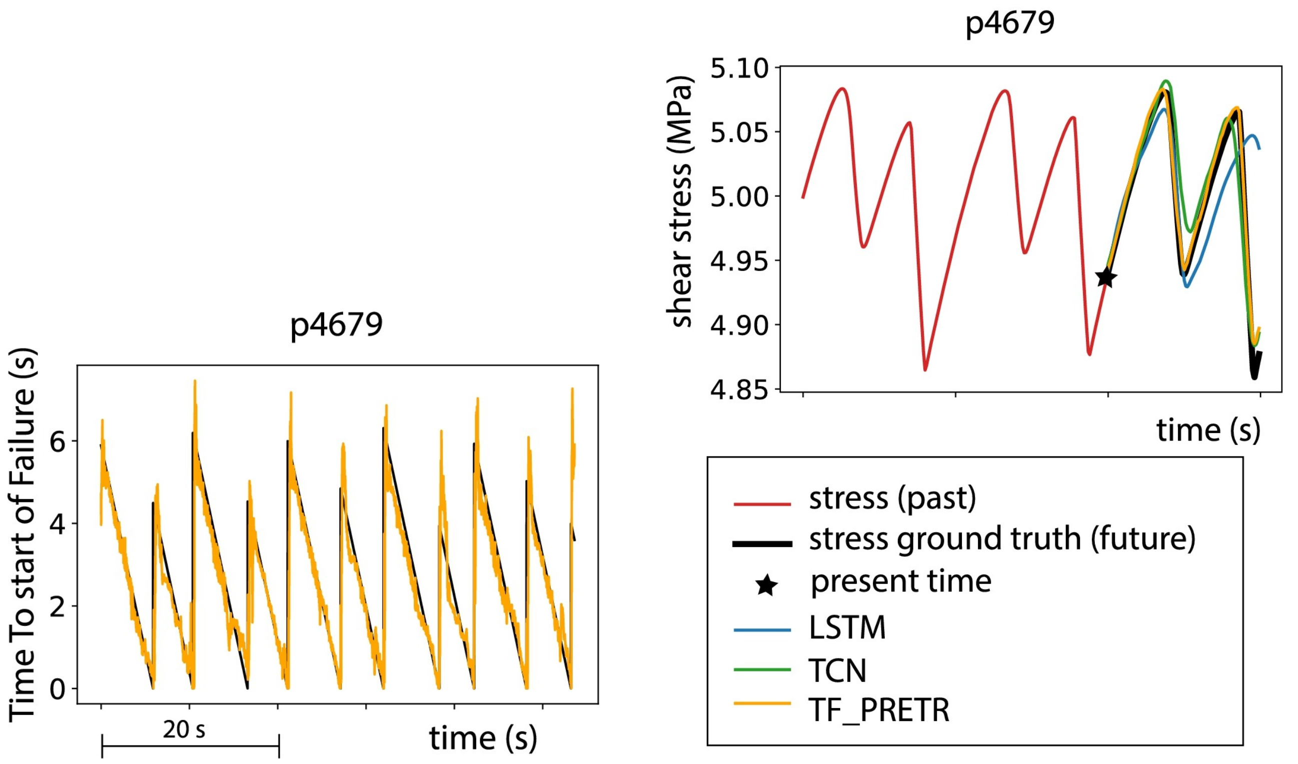
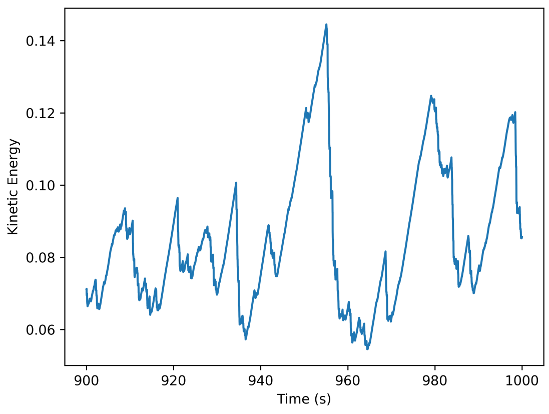
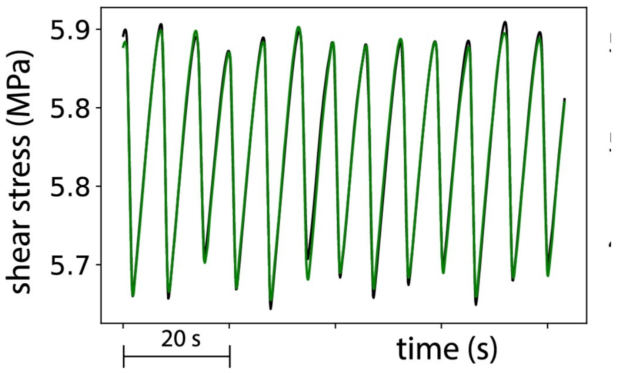


Figure 1: Prediction from a LSTM-CNN model of the time to start of failure of a lab-generated earthquake (a) and forecasting result on the stress during the same experiment (b). (Laurenti et al., 2022)

periodicity makes prediction a relatively easy task, which is likely part of the reason that many machine learning programs are successful. The question then becomes whether the algorithms used in these experiments can also be used on less periodic data. Because the physical situation of real world faults and mechanics behind the earthquakes are often poorly understood, applying these methods directly to real data of ground acceleration recorded by seismographs likely does not lead to useful results. It is important to first get a better understanding of the workings of these methods and what parts of the data they use to make their predictions. In this project, we will apply the algorithms to a dataset with well understood physics without periodicity. The dataset used in this project is artificially simulated data of turbulence. This data is not directly translatable to failures in solid materials, but gives a close proxy to allow us to investigate the machine learning techniques. The results gained from this research hopefully gives a better understanding how these machine learning techniques can be used for real-world applications.



(a)



(b)

Figure 2: Kinetic energy from the turbulence simulation (a) compared to stress from lab created earthquakes (b) (taken from Laurenti et al., 2022). Any instance where there is a decrease in kinetic energy is seen as a failure.

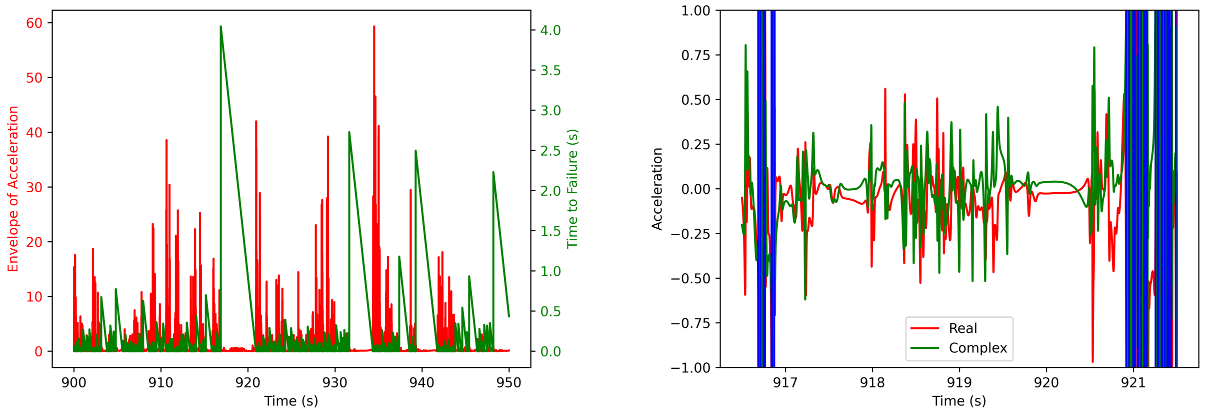
2 Methods

2.1 The data

The data used is a simulation of turbulence created through a shell model of the Navier-Stokes equation (eq. 1) used by Benzi et al., (2022),

$$\delta_t u_n = i(k_n u_{n+2} u^*_{n+1} - \delta k_{n-1} u_{n+1} u^*_{n-1} + (1-\delta) k_{n-2} u_{n-1} u_{n-2}) - \nu k_n^2 u_n + f_n \tag{1}$$

where u denotes particle velocity, k_n is the wave number, ν is the viscosity, f_n is the forcing term and $\delta = 0.4$. From this equation the acceleration (real and imaginary) is calculated. This data is used because it is complicated and non-linear, but the physics is well understood. The acceleration data also provides a good comparison to the acoustic emissions measured in lab-quakes and the kinetic energy can be compared to the stress. The acceleration data consists of a total of 40 different time series, with the first dataset containing the data made with the lowest wave numbers (or the longest periods) and the last dataset containing the shortest periods. The separate shells allow us to investigate if certain wave periods contain more useful data for prediction of failures compared to others, as well as using the total combined data to see if the models are able to identify the relevant data even among a lot of less important information. The full combined acceleration data is calculated by summing the 40 acceleration time series from the separate shells together. From this combined acceleration data, the kinetic energy is calculated. Figure 2a shows an example of the kinetic energy curve. Compared to figure 2b, which shows stress measured in a lab earthquake experiment, the kinetic energy follows a much less periodic pattern. Failures are defined as any instance where there is a decrease



(a) (b)

Figure 3: Input acceleration data (red) compared to output time to failure data (green) (a); Acceleration data during a long inter-failure period (b). Failures are noted by the vertical blue lines. Information between the failures is present but at much lower amplitude than the acceleration data during failures.

in kinetic energy. This definition means that very small, seemingly insignificant decreases in kinetic energy are considered failures just as very large decreases in kinetic energy. After the times of the failures are determined, the time to failure is calculated. In total, the data consists of a total of eight variables: acceleration (real and imaginary), kinetic energy, momentum, time to start of failure, time to end of failure and velocity (real and imaginary), though in this project the momentum and velocity data series are not used. Each time series consists of 20.000.000 data points with 10.000 points per second. The priority of this project is to get a better understanding of the techniques for predicting the time to failure based on acceleration data, but the techniques will also be used to try to forecast future kinetic energy data based on past kinetic energy data. Figure 3a shows an example of input acceleration data and the desired output time to failure data. Figure 3b shows a zoomed in section of the acceleration data during a long inter-failure period. As can be seen from these two figures, the amplitude of the acceleration is much more significant during failures than during the inter-failure period, but there is still a lot of information present between two failures at much lower amplitude.

All model training is done on the Eejit cluster computer of the geosciences department at Utrecht University. This cluster computer consists of 89 nodes with two AMD EPYC 7451 24-Core Processors and 256 GB RAM per node.

2.1.1 Prediction of Time to Failure and Forecasting of Kinetic Energy

Machine learning techniques often struggle to extract patterns in a single, uninterrupted time series as long as these ones. Therefore, the input acceleration data is split into segments of varying size with 90% overlap. The tested segment sizes are 5.000, 10.000, 16.000 and 20.000

points. For each segment a single associated time to failure value is determined. The position of this value respective to the segment is also varied, with the time to failure value taken at the middle of the segment, three-quarters of the segment and at the end of the segment. For the first two, this means that the model uses some information on future acceleration to make the time to failure prediction. This is generally not a problem though, as the focus of this project is on the longer inter-failure periods, which can be close to 100.000 data points long; much longer than the 10.000 data points the largest segment will look into the future. Since the time to failure is linear, as long as the failure is not included within the segment itself the calculation to find the time to failure value at the end of the segment is a simple one.

The time to failure data is adjusted in two ways to test the influence of certain characteristics on the prediction accuracy. The first is by not considering the smallest decreases in kinetic energy to be failures in order to potentially decrease the amount of false positives and prioritise the prediction of large failures, the second by considering multiple failures that occur within a small amount of time from one another to be the same failure. Failures are not instantaneous and the acceleration remains significant in amplitude for a certain period after the start of the failure, with the total time depending on the magnitude of the failure, so if two failures occur quickly after one another, there is no rest period identifiable in the acceleration data. Filtering the acceleration data on the frequency is also done. Some frequencies might contain more useful information than others. By filtering, the algorithms can focus on the useful frequencies. For the filtering of the acceleration data Chebyshev filters are used. For all models that predict time to failure, Random Forest, Temporal Convolutional Networks and Long Short-Term Memory are used.

In lab-quakes, besides prediction of failures based on acoustic emissions, machine learning has also been used to forecast future stress based on past stress. The equivalent to stress in this dataset is the kinetic energy. Past kinetic energy data is used to forecast future kinetic energy. The training for these models is very resource intensive and takes a lot of time. Various input and output lengths are tested to find a preferred input length which balances accuracy and computational time, as well as to find out how fast the accuracy drop-off is in the output data. If the forecasting shows some success, it will give another way of predicting the time of the failures, as well as potentially give insight into the magnitude of said failures. For this problem, Temporal Convolutional Networks and Long Short-Term Memory are used.

2.2 Machine learning techniques

Several different machine learning techniques are used: Random Forest (RF), Temporal Convolutional Networks (TCN) and Long Short-Term Memory (LSTM), with the latter in some cases combined with a Convolutional Neural Network (CNN). The first step of the machine learning process is the training phase. For the prediction of time to failure, the algorithms are given input acceleration data in the form of the envelope of the acceleration, described in equation 2

$$envelope = \sqrt{acr^2 + aci^2} \tag{2}$$

where *acr* and *aci* are the real and imaginary parts of the acceleration respectively. During the training phase, along with the acceleration data, the models are also given output variables, in this case time to failure. The algorithms then try to find a best fitting model to connect the acceleration data with the time to failure data. After the training phase, the models are tested on data not used during training. Now, the models only receive acceleration data. The output of the models is a prediction on the time to failure, which is compared to the known true values of the time to failure to determine the accuracy of the model. The accuracy is determined by using the R^2 value, where a value of 1 means that the model is able to create a perfect prediction, while a value below 0 typically means that the model has no predictive qualities. Besides this score, each model is also evaluated by a visual examination of the prediction compared to the true data, as a model might be accurate for predictions on some parts of the data and less accurate for others, which might not be reflected in the R^2 score. The overall machine learning process for the forecasting of kinetic energy is very similar, though for this problem the input data is past kinetic energy while the output data is future kinetic energy. In most cases, only part of the data set ($\sim$5 million data points) is used to train the models. This provides enough information to the algorithm to train the models while keeping the training time and resources low. The most successful models are then retrained with a higher amount of data to see whether additional data can improve accuracy or not.

2.2.1 Random Forest

The first method investigated is random forest. Random forest was first developed by Ho (1995). A random forest algorithm consists of many decision trees, where each tree has many decision nodes where the path is determined by the input values. Figure 4 shows the general architecture of a Random Forest model. An individual decision tree can be very complex and deep, learning very uncommon patterns and can thus suffer from overfitting to the training data (Hastie et al., 2009). Therefore, many different trees are trained on slightly different data from each other, so that the time series is considered from many different perspectives and criteria. The final output is calculated as an average of the outputs of the individual trees. This averaging method means that Random Forest can be very good with common patterns, but might struggle with extremes. Several different versions of this model exist, of which a selection is used in this paper.

All random forest methods shown in this paper use the scikit-learn package for python (Pedregosa et al., 2011). This package includes several different RF regressors, each with different advantages and disadvantages. The investigated regressors are explained below:

RandomTreesRegressor: This is the most basic random forest regressor. This regressor consists of several decision trees. Every decision tree consists of branching nodes. At each node, a decision is made depending on the input, until no more decisions need to be made and a "leaf" is reached. Each tree is trained in slightly different ways and on slightly different subsets of the data, so that the model is well generalised. The final result is reached by taking the weighted

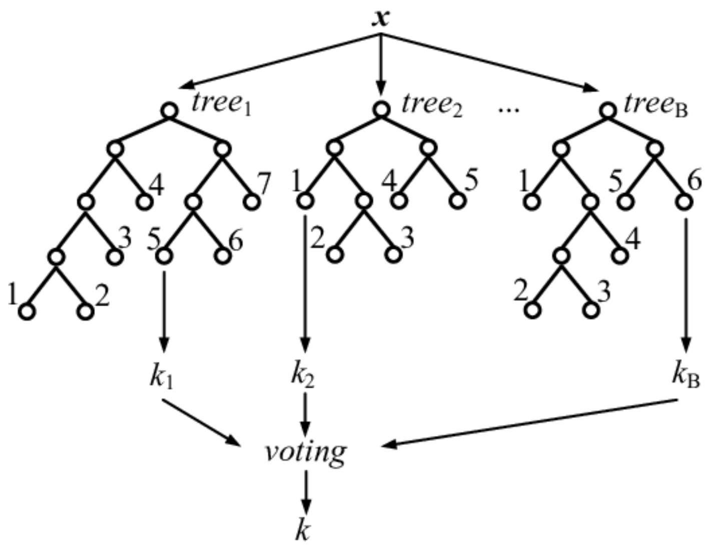


Figure 4: General architecture of a random forest model, with x as input and k as output. Multiple different trees, all based on slightly different data or input arguments so that when averaged, a well generalised prediction is made. (Figure taken from Ryan, 2020)

average of many different trees, where some trees are more important than others. This process can be parallelised relatively easily, which means that it is a very quick method.

ExtraTreesRegressor: This regressor is very similar to the RandomTreesRegressor, but it has more variation in the trees. The sub-samples and validation split of the data are chosen randomly. This creates more diversity between trees, which can result in a faster to train and more accurate model.

GradientBoostingRegressor: In this regressor, trees are not constructed concurrently, but one after another. Each new tree tries to correct and minimise the error of previous trees. This process is thus much slower than the previous two methods, but can potentially be more accurate as the error should be minimised much better if sufficient trees are used. This method also has a variant which is histogram based, which works much more efficiently for large data sets.

2.2.2 Temporal Convolutional Network

The next method is very different from a random forest model. Temporal Convolutional Networks were developed by Lea et al. (2016). The method was first developed for action segmentation, i.e. giving a program a video and asking it to segment and classify the actions portrayed in the video. Previously, this was done by first computing low-level features with a Convolutional Neural Network (CNN) and then inputting those features in a Recurrent Neural Network (RNN). TCN aims to combine a CNN with an RNN, preventing the loss of information between steps. The first use of TCN on a time series was done by Bai et al. (2018). Figure 5 shows the general architecture of the method on time series. The method has also found success in the prediction of time series such as weather (Hewage et al., 2020), energy demand

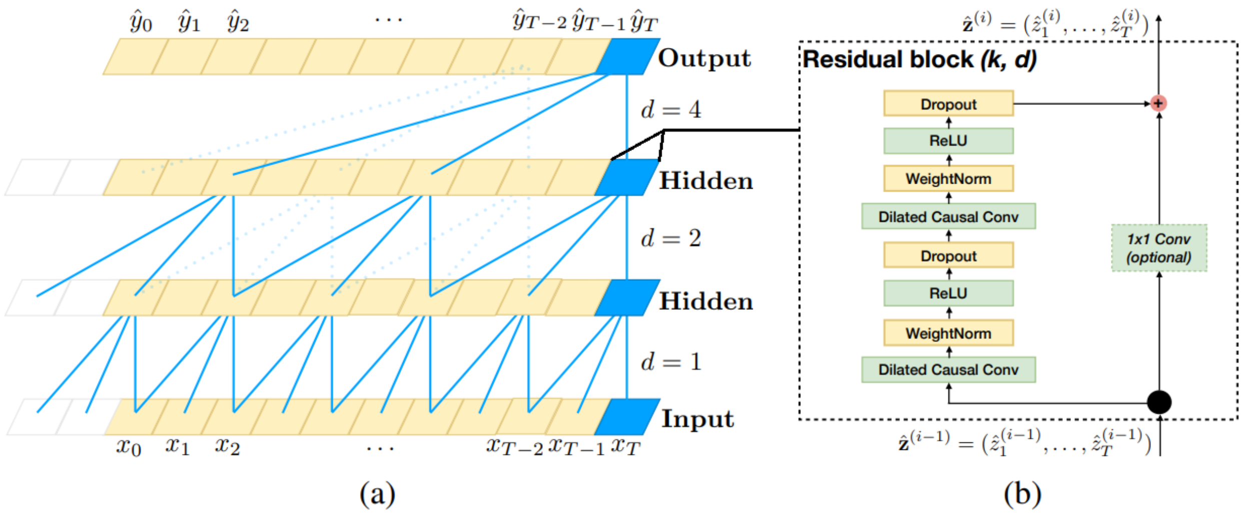

$\hat{y}_0$ $\hat{y}_1$ $\hat{y}_2$ ... $\hat{y}_{T-2}$ $\hat{y}_{T-1}$ $\hat{y}_T$
Output
$d = 4$
Hidden
$d = 2$
Hidden
$d = 1$
Input
x_0 x_1 x_2 ... x_{T-2} x_{T-1} x_T
(a)
$\hat{\mathbf{z}}^{(i)} = (\hat{z}_1^{(i)}, \dots, \hat{z}_T^{(i)})$
Residual block (k, d)
Dropout
ReLU
WeightNorm
Dilated Causal Conv
Dropout
ReLU
WeightNorm
Dilated Causal Conv
1x1 Conv (optional)
$\hat{\mathbf{z}}^{(i-1)} = (\hat{z}_1^{(i-1)}, \dots, \hat{z}_T^{(i-1)})$
(b)

Figure 5: The general architecture of TCN (a), a zoom in on a single residual block (b). The number of layers, the dilation factor (d) and the kernel size (k) among others need to be adjusted for the specific task. (Figure taken from Bai et al., 2018)

(Lara-Benítez et al., 2020) or the stock market (Deng et al., 2019). TCN has shown relative success in the prediction of lab-quakes (Laurenti et al., 2022), and therefore is an interesting method to investigate. The method is not without disadvantages however. It is very sensitive to the input criteria, which makes it hard to fine tune the method to the data. To achieve similar levels of accuracy as the random forest regressors, the method also requires much more time and resources as the random forest regressors, which makes real-time predicting a much bigger challenge. The model described here is made with the keras-tcn package for python (Remy, 2020).

The advantage of this method is the ability to use input data from many time steps ago, as this can often say something about the desired output data. To find the best model for this dataset, many slightly different models are created. The parameters that require fine-tuning include the batch size, number of layers, dilation factor, the optimiser and activation functions, the kernel size and the number of epochs or iterations. The batch size is the number of input points the model will use during training to make one update to the model parameters. Using all points would theoretically result in the best performing model, but it also increases the complexity of the model, reducing efficiency. The dilation factor refers to distance between two input elements. For example, in figure 5 each layer has a dilation factor of 2^{n-1}, where n is the layer number. The dilation factor increases with every layer, usually following a power law. The dilation factor gives the model the ability to use data from far in the past without losing accuracy on local patterns. The kernel size refers to the number of input data points for each residual block. In figure 5, the kernel size is 3. Often, a larger kernel size will increase accuracy up to a certain amount, but will also increase training time, the required RAM and complexity of the model.

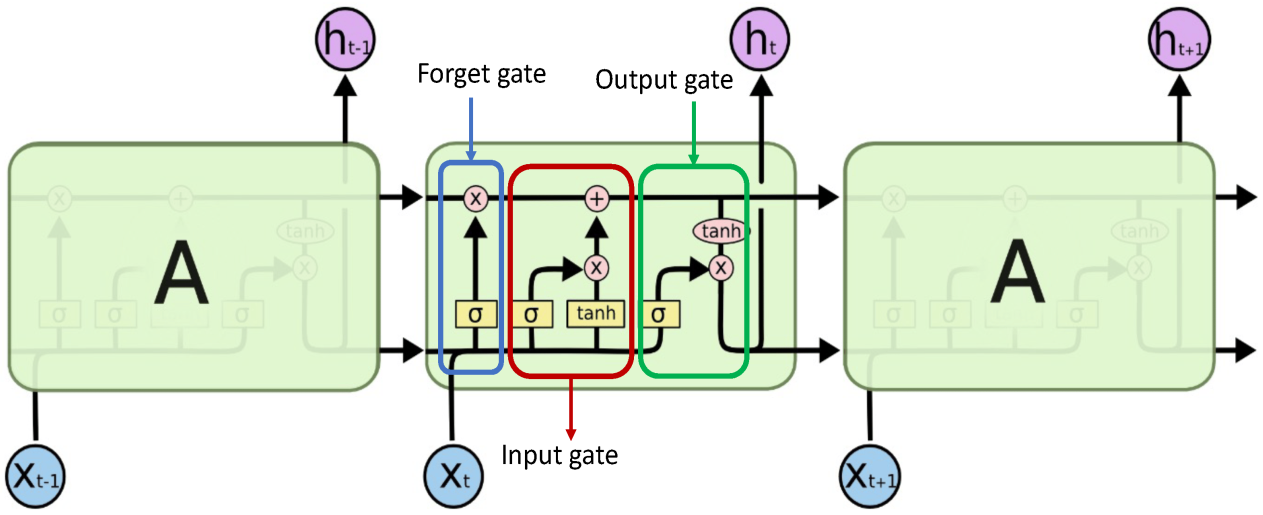


Figure 6: The general architecture of an LSTM cell. X is the input, h is the hidden state and output, σ represents the logistic sigmoid function and tanh represents the hyperbolic tangent activation function. Figure taken from Syed & Ahmed (2023).

2.2.3 Long Short-Term Memory

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN), first developed by Hochreiter & Schmidhuber, 1997. A recurrent neural network allows nodes to receive input influenced by the previous output of the same nodes. This so called "internal state" allows RNN's to work well on time series that require information to be stored a long time, such as in speech recognition (Li & Wu, 2015). A standard RNN suffers from the vanishing gradient problem. The vanishing gradient refers to the gradient of the loss function with respect to the model parameters. If this gradient becomes very small, updates to the model become less significant which slows down or even completely stops convergence of the model. LSTM aims to prevent the vanishing gradient problem by allowing values to flow from one cell to the next without passing through another activation function, preventing the gradient from exponentially degrading. Figure 6 shows a diagram of the architecture of LSTM. LSTM also includes a "forget gate". This forget gate can analyse data from previous time steps and determine whether the data is still relevant to the current time step or not. If not, the data is discarded. This is an advantage as the algorithm needs to handle much less data at one time, making training relatively quick compared to TCN. In this project, LSTM will be used on its own as well as in combination with a Convolutional Neural Network. CNN's are very efficient at detecting local patterns in data, while LSTM is better at long-term patterns. The combination hopes to combine the best parts of both methods. The model code is made with the Keras package for python (Chollet, 2015).

2.3 Data Analysis

To be able to make accurate predictions, it is important that the algorithms receive the correct data. Usually, these machine learning techniques are a bit of a black box in understanding which parts of the data they use and what patterns they detect in the data. Data analysis hopes to give us a better understanding of what the most relevant data is, and how the models make their prediction. For this, analysis on the variance and kurtosis of the acceleration data is done, as well as a HAVOK analysis on the raw acceleration data. The variance and kurtosis are investigated as both have been thought to contain valuable information for the prediction of earthquakes (Rouet-Leduc et al., 2017; Chen et al., 2017). The variance and kurtosis are also used as input in a model for the prediction of time to failure, to see if these two parameters contain relevant predictive information.

2.3.1 Principal component analysis

To get a better understanding what the models use to base their predictions on, the variance and kurtosis of the data is calculated over windows of 16.000 points with 90% overlap. The variance and kurtosis are then also used to make a time to failure prediction. These predictions are slightly less accurate than using the raw acceleration data, but are also much faster as they use much less data points for training. Further analysis on the variance and kurtosis also allows us to better investigate the models, to find out which parts of the data the models use the most to make the predictions.

Principal component analysis (PCA), a form of linear dimensionality reduction, is applied following the method of Ali et al. (2019). The PCA is applied to the variance and kurtosis separately to investigate the importance of both parameters. PCA hopes to find repeating patterns in the data as well as outliers. For this, the time series of the variance and kurtosis are first put in a Hankel matrix (equation 3, where v is the variance or kurtosis, x is the matrix width and y the number of rows). Every row of the Hankel matrix thus contains part of the variance or kurtosis time series, with every row identical to the one above it but shifted by one time step. Then, singular value decomposition is applied to the matrix to obtain the principal components. The width of the Hankel matrix is highly influential on the results and thus needs careful calibration. Each row needs to be long enough to contain any potential patterns visible in the data, but not so long that each row covers multiple medium to long inter-failure periods, as that will make identification of local patterns more difficult.

$$H = \begin{bmatrix} v_0 & v_1 & \dots & v_x \\ v_1 & v_2 & \dots & v_{x+1} \\ \dots & \dots & \dots & \dots \\ v_y & v_{y+1} & \dots & v_{y+x} \end{bmatrix} \tag{3}$$

2.3.2 Uniform Manifold Approximation and Projection

Another method of dimensionality reduction is Uniform Manifold Approximation and Projection (UMAP) developed by McInnes et al. (2018). Like PCA, UMAP looks for recurring patterns and outliers in the variance and kurtosis data. However, while PCA aims to find linear combinations of the variables that capture most of the variance in the data, resulting in mostly global structure, UMAP tries to find both local and global patterns in the data through non-linear methods. It does this by first approximating the manifold on which the data lies followed by modelling this manifold through a fuzzy topological structure. The embedding is then found by looking for a low dimensional projection of the data with the closest possible topological structure to the topological structure on the high dimensional data. This method makes UMAP much more computationally efficient than PCA. Like PCA, the result of the UMAP analysis is dependant on a couple of parameters. These parameters are the Hankel matrix width and the "number of neighbours". The number of neighbours determines by how many other points the position of a specific point is calculated. A low value for the number of neighbours means that the position of a point is only calculated by considering the few data points that are most similar, resulting in a figure showing mostly local structures, while a high value of the number of neighbours results in a figure showing mostly global structures.

2.3.3 HAVOK analysis

The HAVOK (Hankel Alternative View Of Koopman) analysis was developed by Brunton et al. (2017). Koopman posits that any non-linear function can be described by an infinite number of linear functions (Koopman, 1931). Of course, an infinite number of functions is not something that can be handled by a computer system, and thus HAVOK was developed. HAVOK aims to describe a non linear function with $r-1$ linear functions and one non-linear function describing the remainder of the initial function that can not be described linearly. Equation 4 describes the system used in HAVOK

$$\frac{d}{dt}\mathbf{x}(t) = \boldsymbol{A}\mathbf{x}(t) + fx_r(t) \tag{4}$$

where $\mathbf{x}$ describes the first $r-1$ linear functions and fx_r is the non-linear remainder, also known as the forcing term. Then, by analysing the magnitude of the forcing term it is possible to determine when significant, non-linear events are happening in the time series (forcing active). In some cases, the non-linear function becomes significant some time before the event actually occurs, basically giving a warning in advance of the event. The HAVOK analysis is done on a part of the acceleration data, in total one million data points long. This length is chosen as it contains enough data to extract patterns while keeping the required resources and computation time low. The kinetic energy, the envelope of the acceleration and the real and imaginary parts of the acceleration separate are all evaluated with the HAVOK analysis. The HAVOK analysis is done on the full combined acceleration as well as on acceleration data of the individual shells.

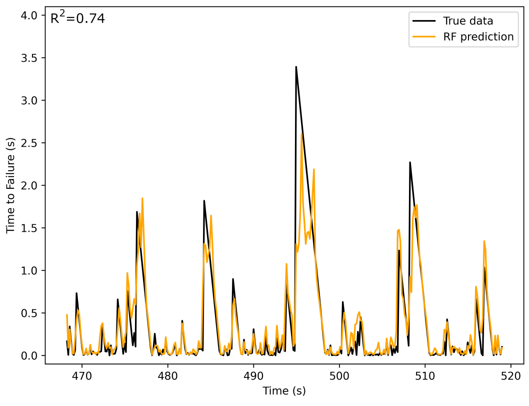
3 Results

3.1 Time to failure prediction

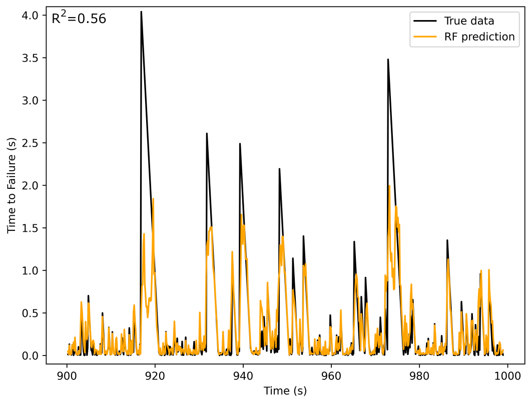
3.1.1 Random Forest

The different random forest methods all managed to create a decent prediction model. Overall, the HistGradientBoostingRegressor and the ExtraTreesRegressor were the most successful, with the latter being slightly more accurate and also requiring much less resources and training time. Both models achieved an R^2 score of around 0.75, though these scores are highly dependant on the data in the testing window. Figure 7 shows two predictions made with the ExtraTreesRegressor and a segment size of 16.000 points, where the prediction accuracy of high times to failure between the figures is dependant on the input data used and the amount of high frequency acceleration data present in inter-failure periods. The models are very good at correctly predicting lower times to failure, but do not perform very well in correctly predicting time to failure higher than two seconds. So, if there are many instances of such long times to failure in the testing data, the performance scores will be relatively low. The RF algorithms also suffer from overfitting. If the model is asked to give a prediction on the basis of acceleration data that has also been used during training, the model manages to provide a perfect prediction.

The model described above was trained on a total time series length of five million data points. Using more data in training does not lead to a more accurate model. The exact reason for this is not entirely clear, but it is potentially due to the model becoming too complex. The models where the time to failure is taken at the centre of the segment perform best. These models combine the strength of receiving "future" data and still get enough past data to make a proper prediction. The segment length has significant effect on the performance of the models. Figure 7a shows a model trained on data segmented into segments of 16.000 data points while figure 8 shows models trained on other segment sizes. The best segment size that balances performance, training time and the amount of future data points received is a segment length of 16.000 data points long. Models trained with shorter segments perform significantly worse at correctly predicting higher times to failure, while a model trained with segments of 20.000 data points performs roughly equal, but receives a lot more "future" data, thus decreasing how far ahead of time the failures can be predicted.



(a)



(b)

Figure 7: A Random Forest model trained with the ExtraTreesRegressor. Two different testing windows also show different accuracies.

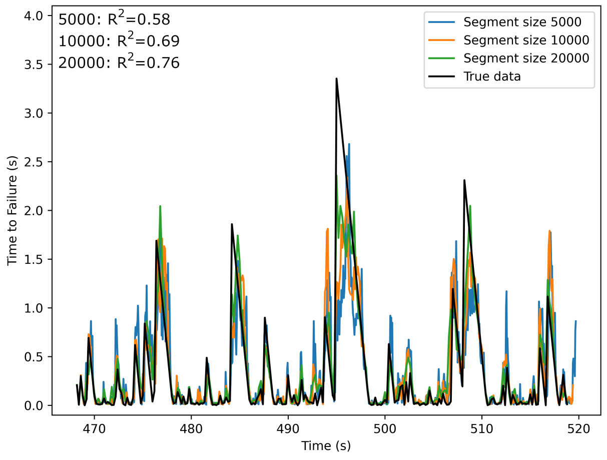
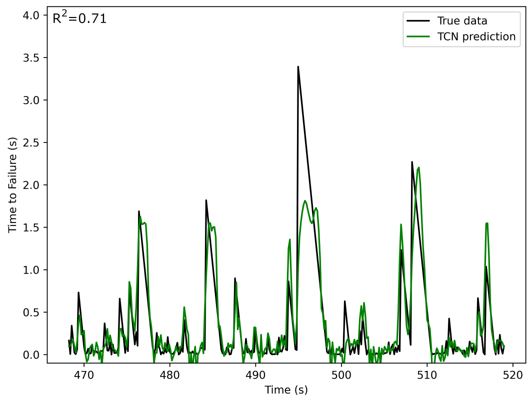


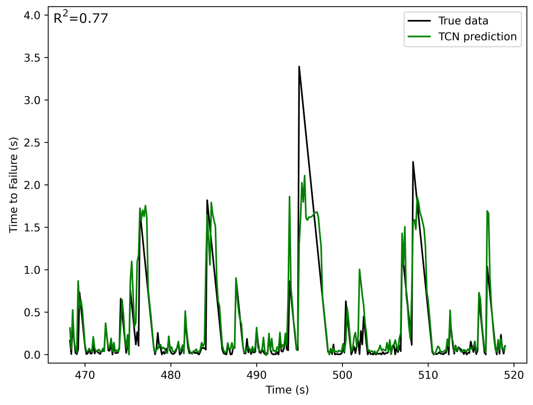
Figure 8: Random Forest models trained on differently segmented data.

3.1.2 Temporal Convolutional Networks

Overall, the Temporal Convolutional Networks perform very similarly to a random forest model. However, TCN requires much more training time. While RF is able to complete training and testing on a time series of 5 million data points in around 5 minutes, TCN takes anywhere between 40 minutes to 24 hours to complete, depending on the parameters. The ExtraTreesRegressor for random forest is very well parallelisable as each tree can be trained on their own processor, making calculation much more efficient than TCN. If the exact same input data as used in the Random Forest models is used, a maximum R^2 score of around 0.7 is reached. However, if 95 percent of data points per segment are discarded, so that each segment now consists of just 800 data points with a much larger time delta between points, performance increases to a maximum R^2 score of 0.77. This reduction of input data greatly reduces the needed resources and training time and decreases the complexity of the model, improving overall performance. With this reduction in data, the ideal parameter settings for training are a batch-size of 100, kernel size of 10 and dilations 2^{n-1} where n is the layer number. A total of 12 layers were used. More than twelve layers does not result in a better performing model. Less layers reduces the required training time and computational resources, but result in a less accurate model. Figure 9 shows the prediction of time to failure of several TCN models compared to the real data. TCN still struggles with accurately predicting high time to failure, just like RF, though it is a bit more consistent in its prediction of medium times to failure than RF. Unlike RF, TCN does not suffer from overfitting. If the model is asked to predict time to failure on acceleration data seen during training, the performance score is equal to its performance on unseen data. The influence of the segment size and location of the time to failure choice within that segment was found to be the same as for RF.



(a)



(b)

Figure 9: Two models trained with TCN. (a): A model trained on the full acceleration data. This model is good at predicting medium times to failure, but fails with high time to failure. (b): A model made with the best performing parameters of previous tests, using just 1 out of every 50 data points.

3.1.3 Long Short-Term Memory

The overall accuracy of the best LSTM model is slightly lower than those of RF and TCN with an R^2 value of 0.69. This loss of accuracy is mostly due to a worse prediction of high time to failure and low time to failure, while the prediction accuracy at medium time to failure is very comparable between LSTM, TCN and RF. LSTM is relatively fast to train, taking roughly the same amount of time as the ExtraTreesRegressor for random forest. Figure 10 shows the predictions made by a model without convolutional layers and with convolutional layers. For the second model, two convolutional layers were found to provide the best performance, though the improvement is very minimal and only significant for very low time to failure prediction accuracy. More than two convolutional layers does not improve performance and just increases the necessary training time and computing resources. The LSTM model also does not suffer from overfitting, with prediction scores on data seen during training very similar to scores on unseen data. Once again, the influence of the segment size and location of the time to failure choice within that segment was found to be the same as for RF and TCN.

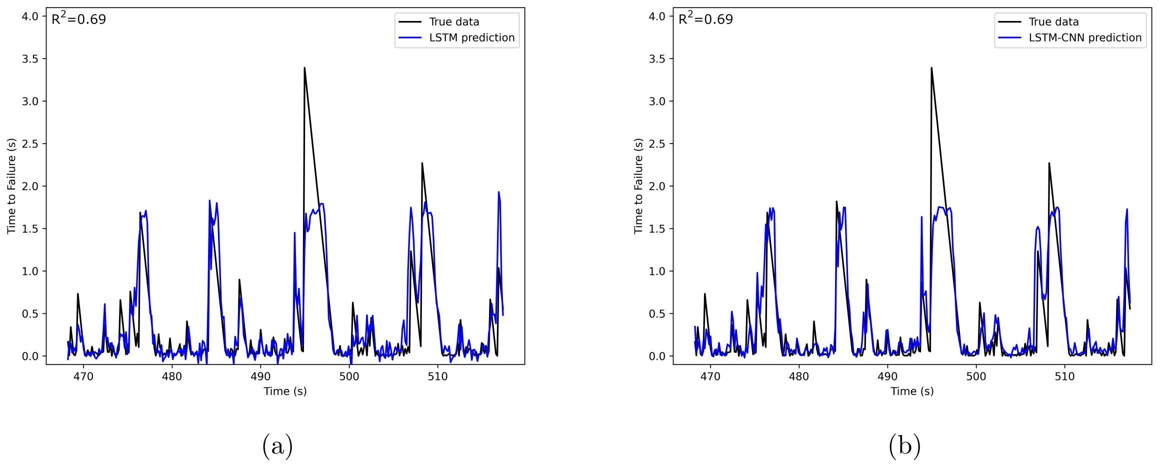
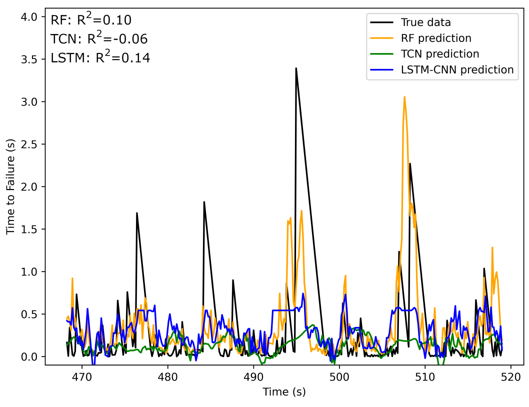


Figure 10: Predictions made by an LSTM model without convolutional layers (a) and with convolutional layers (b).

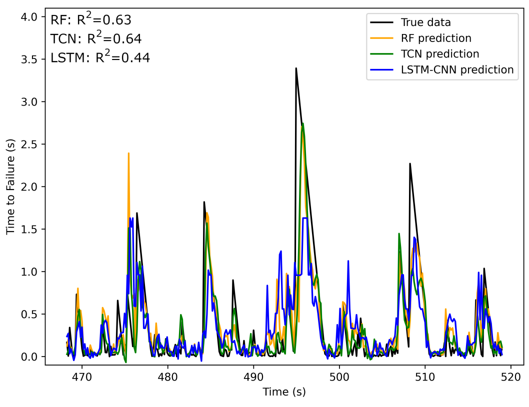
3.1.4 Application of Techniques on Individual Shells

The complete acceleration is composed of the sum of 40 frequency time series. Dataset one consists of only the longest periods (low wave frequencies), while dataset 40 only contains the data with the shortest periods. Fifteen of these shells contain very little energy and are unlikely to contribute much to the prediction of time to failure and therefore only 25 of the total 40 shells are investigated in further detail. These 25 separate time series are first subdivided in 3 groups, with the first group containing shells with long periods, the second group containing shells with medium periods and the third group containing the rest (short periods). Each of these groups were then used to train models in the same way the full acceleration time series was used.

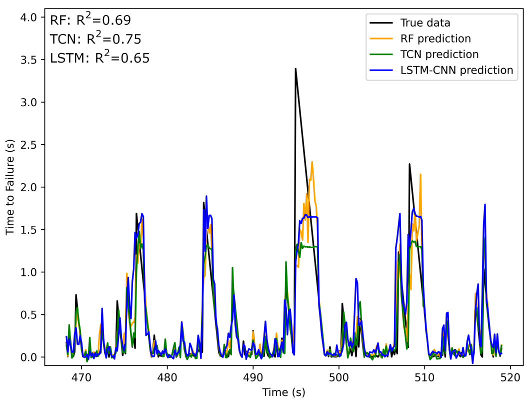
Figure 11 shows the prediction results of the models trained on the shell groups. Overall, the models trained on the shell group with the shortest periods performs best considering the R^2 score. However, the RF and TCN models trained on the middle shell group give a more accurate prediction for high time to failure. The overall R^2 score is lower as the accuracy at low time to failure is much lower. The models trained on the shell group with the longest periods



(a)



(b)



(c)

Figure 11: Prediction results from the three different algorithms on shell groups with long periods (a), medium periods (b) and short periods (c).

do not give accurate predictions. The TCN model has a negative R^2 score, implying that this model has no predictive ability. The RF and LSTM models perform slightly better, but are still highly inaccurate.

Following these results, the different machine learning techniques are also applied to each individual shell. Figure 12 shows the same inter-failure period as figure 3b but now with only specific periods plotted. The shells with the absolute longest periods (shell 1-5) are not able to create predictions with any significant degree of accuracy (figure 13a). These shells contain wave periods that are often longer than even the longest inter event time spans and the failures themselves are hard to recognise in the acceleration data (figure 12a). The models trained on shell 6-9 show dramatic improvement of prediction accuracy on long time to failure (figure 13b). These shells contain periods that are shorter than the problematic long inter-failure periods and

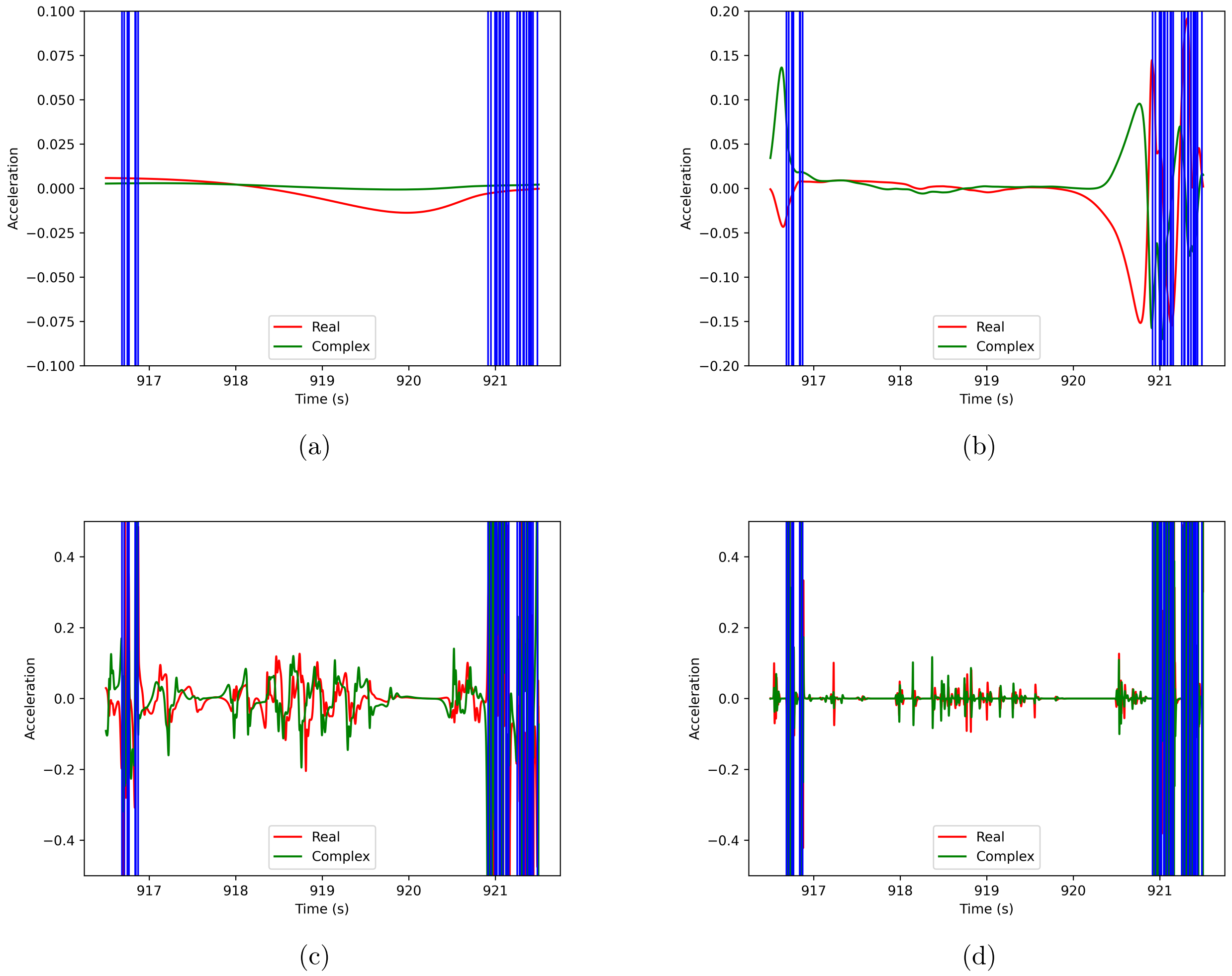
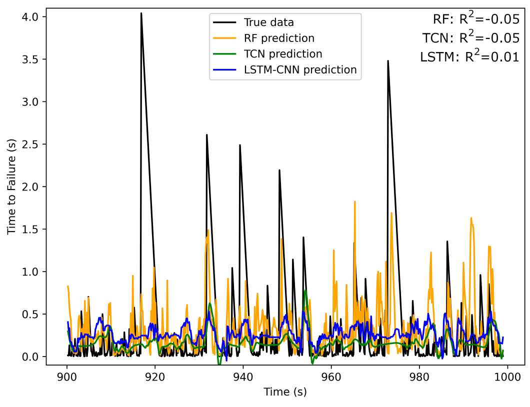
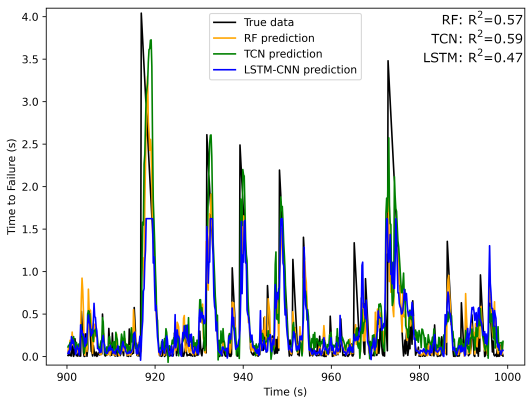


Figure 12: Acceleration data of individual shells during a long inter-failure period, with failures noted by vertical blue lines. (a): shell 2, (b): shell 8, (c): shell 14, (d): shell 20. Shell 2 shows almost no significant data and the failure is not well recognisable in the acceleration data. In shell 8, there is a significant amount of data and the failures are recognisable in the data. Shell 14 and shell 20 both clearly show the failures, but also have significant amounts of high frequency data in the inter-failure period.

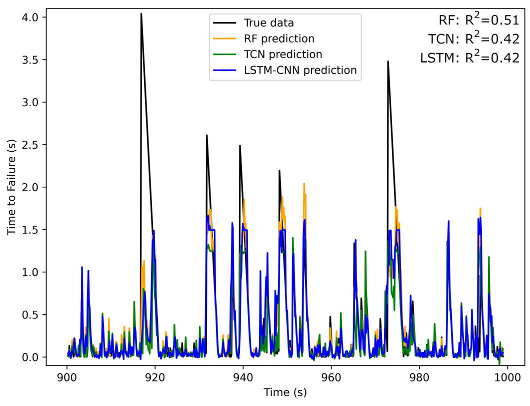
where the failures are well recognisable in the data, while also not including a lot of short period inter-failure information (figure 12b). Shells with even shorter periods do not perform well in the prediction of high time to failure (figure 13c). The acceleration data of these shells might show a lot of high frequency data in long inter-failure periods (figure 12c) similar to the wave patterns found during failures, and the shells with the highest frequencies contain almost no relevant data in inter-failure periods for failure prediction (figure 12d). However, as figure 13d shows, there is still useful information present in these high frequencies as the overall accuracy



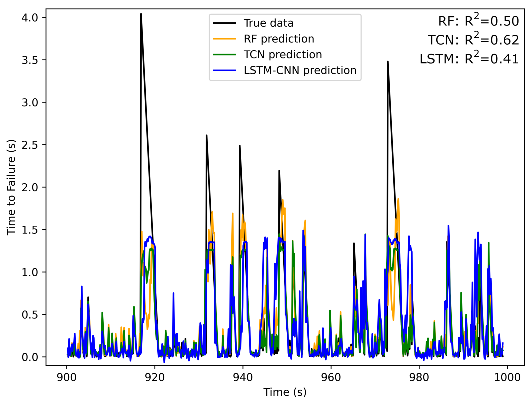
(a)



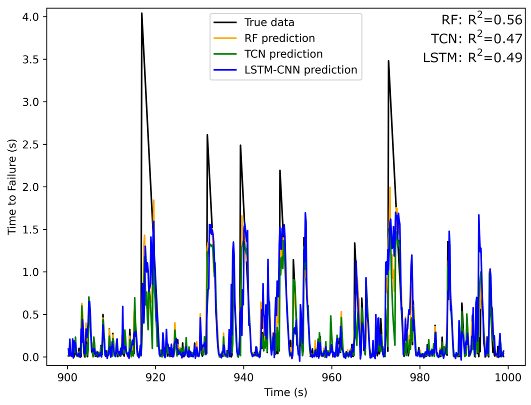
(b)



(c)



(d)



(e)

Figure 13: Prediction results of models trained on acceleration data only from shell two (a), shell eight (b), shell 14 (c) and shell 20 (d), compared to prediction results of models trained on the full combined acceleration dataset (e).

of the model trained on this shell is relatively good.

For prediction of higher times to failure, shell 8 provides the most valuable information as can be seen in figure 13b. Especially the TCN model performs very well in the prediction of higher times to failure at a cost of lower time to failure prediction accuracy. The RF model also shows significant improvement in the prediction of high times to failure while the LSTM model shows very little improvement.

3.1.5 Filtering of Data

Several different methods of filtering were investigated to see whether the removal of some information could improve performance. First, the threshold for what is considered a failure is raised so that not every single decrease in kinetic energy is considered a failure anymore. The goal of this is to hopefully decrease the amount of false positives, as acceleration signatures of the failures now need to be significantly higher in amplitude than the inter-failure high frequency information to be considered a failure. Figure 14 shows the predictions with this adjusted time to failure data. The prediction accuracy of these models is lower than the accuracy of models where every decrease in kinetic energy is seen as a failure.

Following this, instead of removing the smallest failures, failures occurring within a very short time from one another are considered the same failure. This is done as the previous failures might be the direct cause of the next failure, as the magnitude of the acceleration is still very significant for a certain period after a failure. For the models to work properly using this adjusted time to failure data, the time to failure is only considered from the final failure in a failure sequence. The time to failure is set at a constant zero during a failure sequence (figure 15). In this figure failures that are occur less than 0.5 seconds from one another are considered to be part of the same failure sequence. This method of adjusting the data results in a very small improvement in the prediction of medium times to failure compared to figure 7a. The model has no improvement in the prediction accuracy of very high time to failure. Changing the threshold so that the inter-failure period needs to be even higher for them to be considered separate failures does not lead to a more accurate prediction.

The final method of data manipulation is on the acceleration data instead of the time to failure data. As much of the inter-failure data is very high frequency in form, it might be possible to filter this information out without losing useful lower frequency information, preventing the models from falsely identifying the high frequency data as failures. In theory, this is quite similar to using only shells with specific periods, though in this case the frequencies are filtered out of the full combined acceleration with Chebyshev filters. The exact frequencies that need to be filtered out are unknown, so several filters are tested. Figure 16 shows the results of models trained on filtered acceleration data. No method of filtering was found that is able to match the performance of using just the periods of shell eight in the prediction of high time to failure, though some do come close. The best performing model for the prediction of high time to failure is one trained with data filtered through a low-pass filter which filters out frequencies

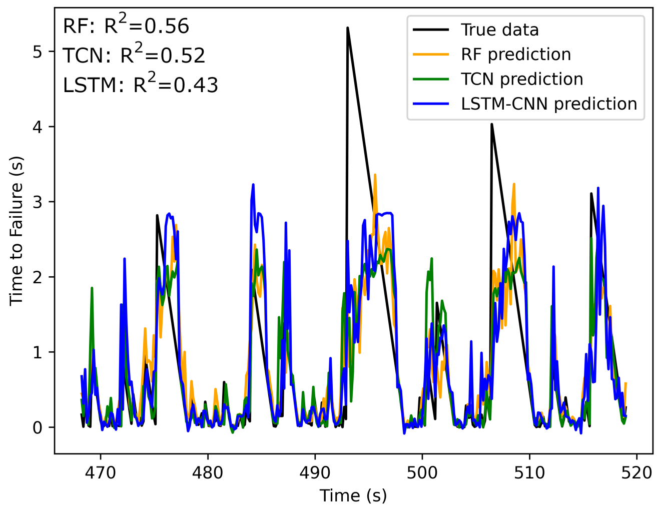


Figure 14: Models trained on data where small decreases in kinetic energy are not considered to be failures. Overall performance is not improved over using a dataset where every decrease in kinetic energy is considered a failure.

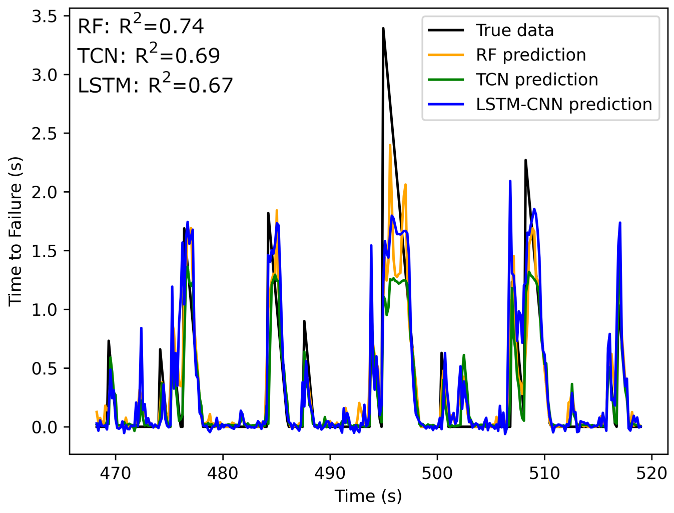
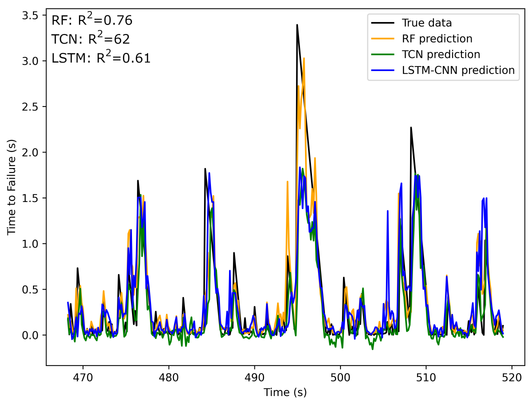
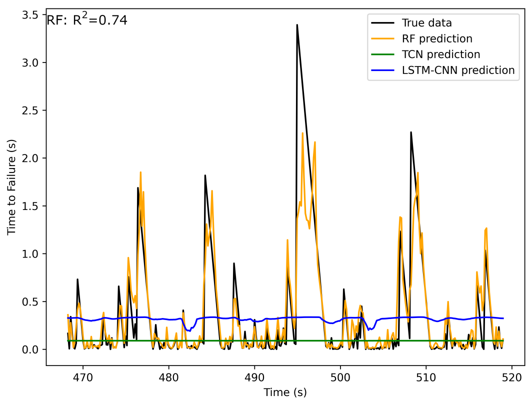


Figure 15: Models trained on data where failures occurring within a very short amount of time from one another are considered the same failure. The time to failure during a sequence has to be set to zero during the sequence for performance reasons, meaning that the time to failure prediction is from the end of the previous failure sequence.

above 90 Hz. The RF model trained on this data is able to quite accurately predict higher times to failure, but is slightly less accurate in the prediction of low time to failure. It is somewhat expected that these lower frequencies also contain the more important information for prediction of higher time to failure as the shells that contain the most useful information for prediction of higher time to failure (shell 6-10) also contain frequencies mostly below 90 Hz. TCN and LSTM do not show this improvement in the prediction of high time to failure. Figure 16b shows a figure with models trained on data with a high-pass filter at 1000Hz. In this case, the TCN and LSTM models are completely unable to form predictions, while the RF model is still relatively good. The accuracy of high time to failure predictions is not as good as for the model trained on data filtered through the low-pass filter, but it has great accuracy for medium-low times to failure. This suggests that while there is still useful information present in the high frequency data, it is much harder to extract it, especially for TCN and LSTM.

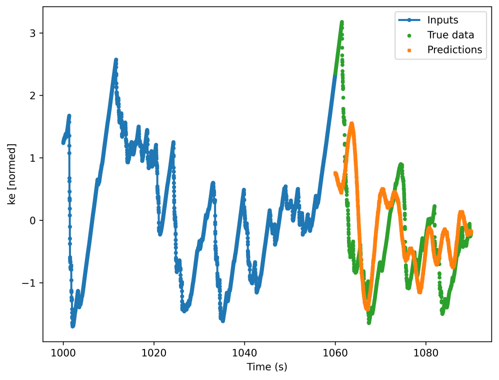


(a)

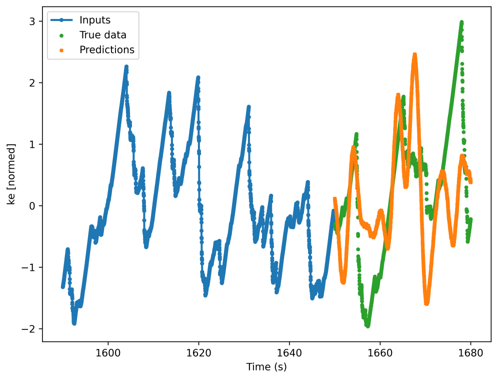


(b)

Figure 16: Two models, trained on data with frequencies above 90 Hz filtered out (a) and data with frequencies below 1000Hz filtered out (b). If frequencies below 1000Hz are filtered out, TCN and LSTM are unable to provide proper predictions.



(a)



(b)

Figure 17: Two examples of forecasting of kinetic energy, done with TCN. The rough pattern can be seen in the output of the TCN model, but the overall accuracy is very low.

3.2 Forecasting of kinetic energy

Overall, attempts to accurately forecast the kinetic energy have not been successful. While the model trains well and achieves great accuracy when applied to data seen during training, no method is able to forecast kinetic energy to a high level of accuracy on data not seen during training. The best performing model is a TCN model which uses only 100 data points per second instead of the 10.000 data points per second available in the data set. In total it uses an input length of 60 seconds worth of data points to forecast the next 30 seconds. This model is still not accurate, but is sometimes able to forecast a kinetic energy curve that follows roughly the same pattern as the real data. Figure 17 shows results of the forecasting done on data seen during training and unseen data. While this prediction sometimes seems to accurately determine the time of a failure, it also misses quite a few failures and has many false positives. The magnitude of the predicted failures is also highly inaccurate. Whether this model is actually able to forecast the rough pattern of the kinetic energy is highly dependant on the starting point of the forecast, though it is not clear in what way this starting point influences the forecast. In some cases, the model is able to forecast the rough pattern if the starting point is in the middle of a long inter-failure period, in other cases it is only able to forecast the rough pattern if the starting is just after a failure. Because of this inconsistency, it is not possible to draw meaningful conclusions from these forecasts or use them to predict the time to failure and its magnitudes.

3.3 Data Analysis

Variance and kurtosis are two characteristics of the acceleration data that might hold useful information for the prediction of time to failure. The variance and kurtosis are calculated over a segments with 16.000 data points with 90% overlap, to remain consistent with the segment size used in the time to failure predictions with the acceleration data. Figure 18 shows predictions made with the variance and kurtosis. Overall, these predictions are much less accurate than using the full acceleration data. The TCN model performs significantly worse than the LSTM and RF models, with the latter still producing somewhat accurate predictions for medium times to failure. The training time required for these model is also much lower than the training time when using the full acceleration data. This does suggest that the variance and kurtosis both contain valuable information, but it is not the only information the models use to make their predictions. Further analysis of the variance and kurtosis is done with PCA and UMAP.

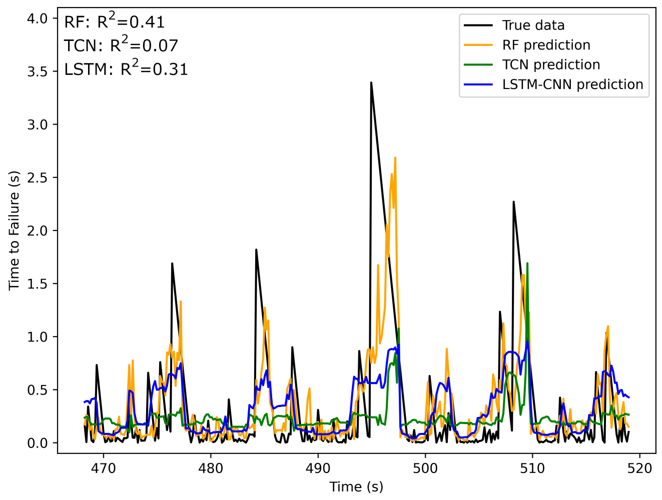
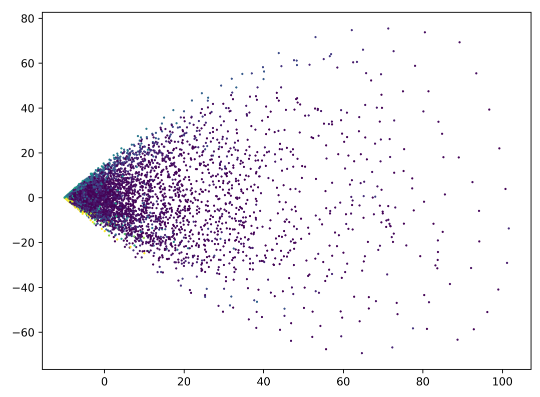


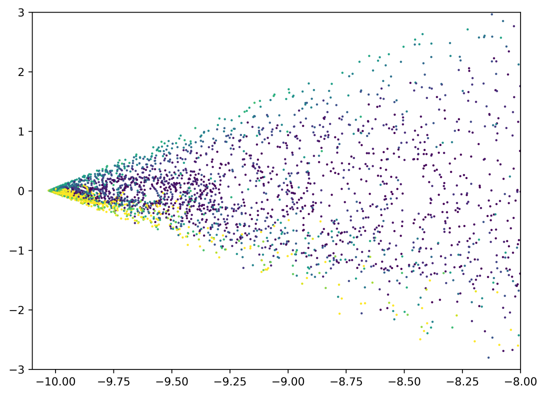
Figure 18: Models trained on the variance and kurtosis of the acceleration data. Overall the accuracy is much lower than models trained on acceleration data, but RF and LSTM still produce somewhat valuable predictions.

3.3.1 Principal Component Analysis

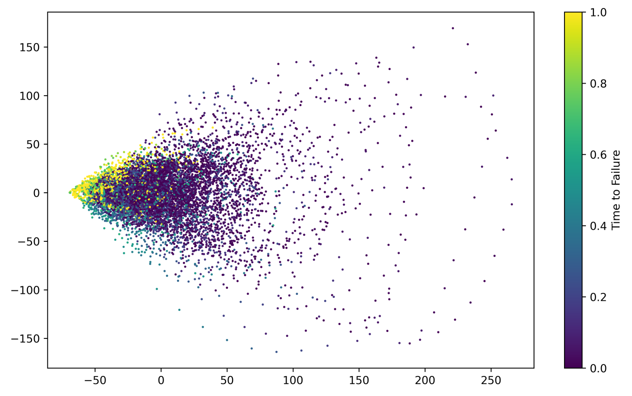
Figure 19 shows a 2D projection of the PCA on variance and kurtosis. Each point in the figure represents a row of the Hankel matrix constructed with the variance or kurtosis data used in the analysis, and is coloured by the corresponding time to failure value of the window. The specific pattern created by the method is dependant on the matrix width. The matrix width has to be be calibrated so that is not significantly higher than the shortest failure cycle, while each window should still contain enough information. As the shortest failure cycles in this data



(a)



(b)



(c)

Figure 19: The first two components (first component on x-axis, second on y-axis) of the Principle component analysis on variance (a), a zoom-in of the PCA on variance (b), and PCA on kurtosis (c) of the acceleration data. The colour denotes the time to failure value associated for each point. A slight concentration can be seen in both the PCA of variance and kurtosis, but many exceptions exist.

set are just 1-2 time steps long (after calculation of variance and kurtosis on the raw data using 16000 data points per calculation), a matrix width of 20 is chosen to make sure that each row of the matrix has enough information for a useful analysis while still retaining the ability to detect local patterns. As a Hankel matrix row contains 20 values, with each variance or kurtosis value calculated from 16.000 data points with 90% overlap, a single matrix row covers 46.400 data points of the original acceleration data. The time to failure value associated with a matrix row is chosen as the point in the middle of this window. The distance from one point to any other point shows how similar the patterns within the rows of the Hankel matrix are to each other. A cluster of points shows the presence of a common pattern present in the data for those points.

As can be seen in figure 19, a cluster of points with high time to failure values can be seen in the PCA of variance and kurtosis, but it is not consistent, as there are many points with high time to failure outside this cluster. If the variance and kurtosis are fully correlated with the time to failure, there should be a clean separation between points with high time to failure and low time to failure. Figure 20 shows the results of the PCA plotted against time, as well as the time to failure. Here, we can again see a correlation between the PCA and the time to failure, with the first PCA component of both the variance and kurtosis having their lowest values during long inter-failure periods. This correlation is more significant for the PCA on variance, while the PCA on kurtosis is not completely consistent in its comparison with the time to failure. Both the PCA on variance and the PCA on kurtosis also show low values during some failure sequences. While there is a correlation between low values of the PCA components during long inter-failure intervals, the PCA components themselves can not be used for accurate prediction of the time to failure. The PCA component does not gradually rise to higher levels as the failure approaches. In some cases, the PCA component rises suddenly in value just before the failure, in other cases the PCA value remains low for a small amount of time after the failure.

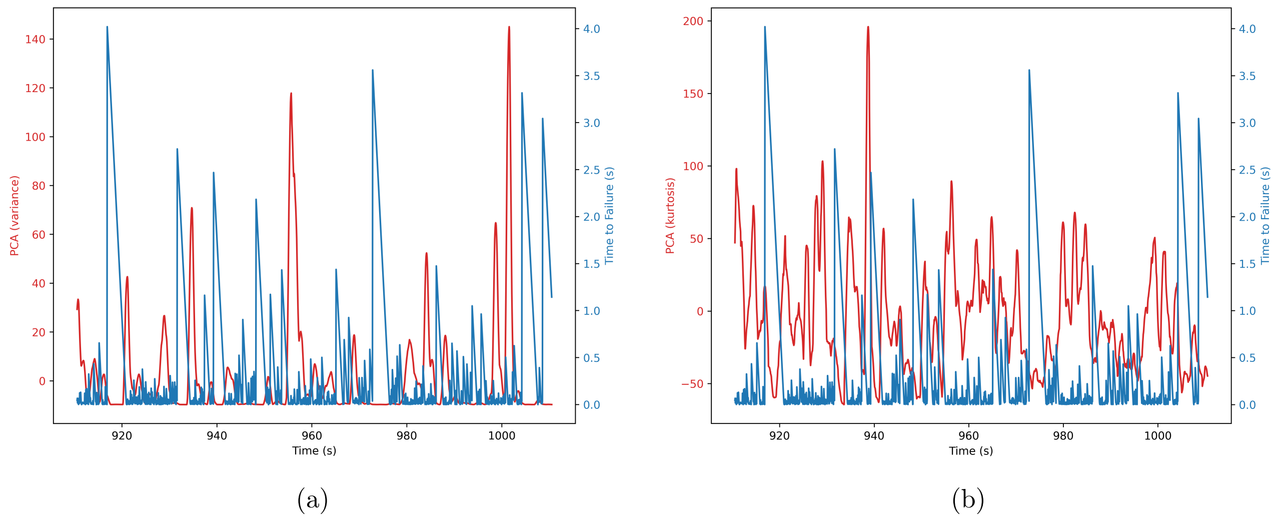


Figure 20: The first component of the PCA on the variance (a) and the kurtosis (b), plotted against time, compared with the time to failure.

3.3.2 Uniform Manifold Approximation and Projection

Figure 21 shows the 2D projection of the UMAP analysis. In this case, we can clearly see a distribution pattern of the data points and the corresponding time to failure in the UMAP on variance. There are still some exceptions, but overall there is a division between low time to failure and high time to failure data points. The UMAP analysis on kurtosis does not lead to clean results like the UMAP on variance, but there is still some concentration of points with high time to failure visible. When the results of the UMAP analysis are plotted against time

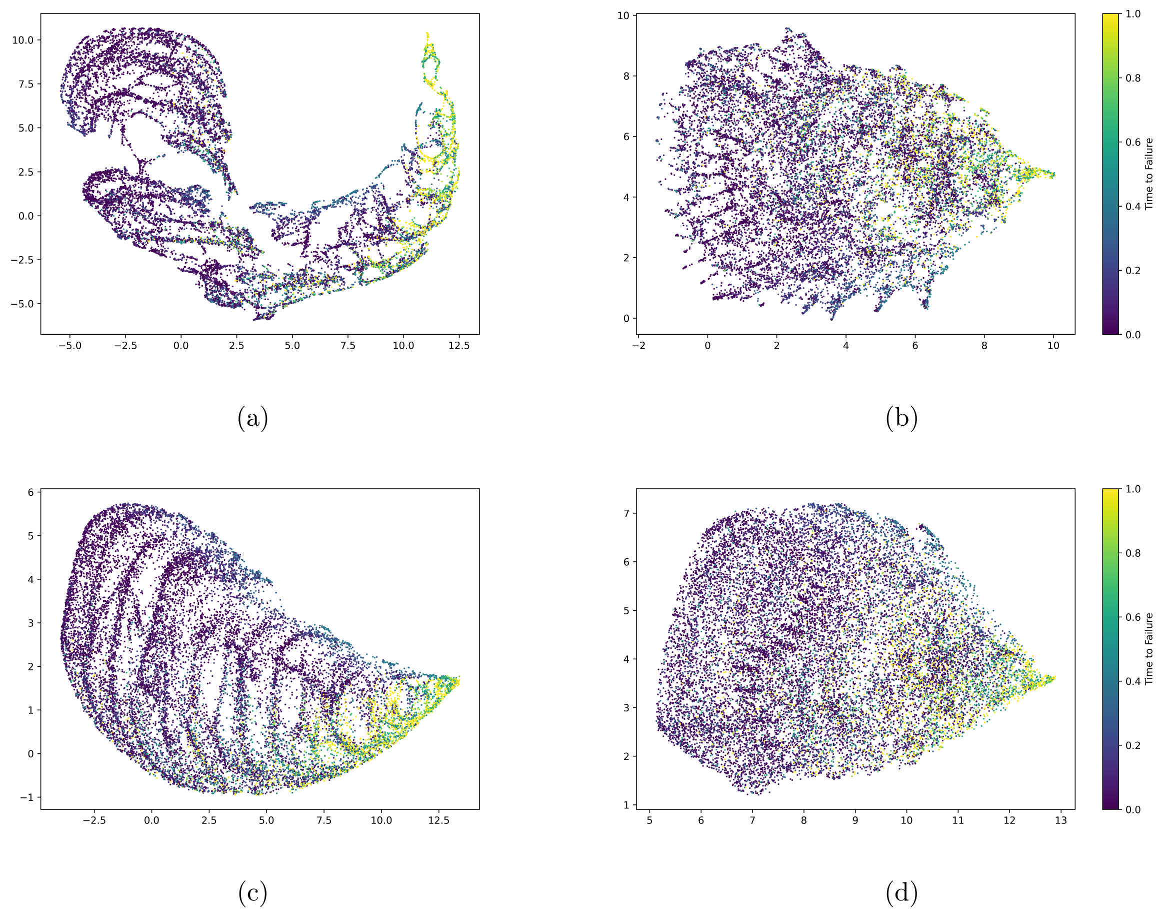


Figure 21: The first two components of the UMAP analysis on variance (a) and kurtosis (b) with a high "number of neighbours" (showing global patterns) and the UMAP analysis on variance (c) and kurtosis (d) with a low "number of neighbours" (showing local patterns). The colour denotes the time to failure value associated for each point. A clustering of high time to failure points can be seen in both the UMAP of variance and kurtosis, but many exceptions exist.

and compared to the time to failure (figure 22), a relation between the UMAP on the kurtosis and the time to failure is visible, with the first component of UMAP having low values during long inter-failure periods and high values during a quick succession of failures. The first component of UMAP of variance does not show consistent correlation with the time to failure. While the first component of UMAP is typically high during long inter-failure periods, it is also high at other times. The analysis with both PCA and UMAP suggest that there are consistent patterns associated with long inter-failure periods. However, both also show exceptions, which is also recognisable in the predictions, where some long inter-failure periods do not contain these same patterns and are much less accurately predicted than others of similar length. UMAP might allow for slightly better predictions based on the first components than

PCA. In figure 22b we can see that while the UMAP component reaches very low values during long inter-failure periods, it starts to recover to higher values significantly ahead of the failure, unlike the components of PCA. UMAP also has the advantage of being able to show both local and global patterns.

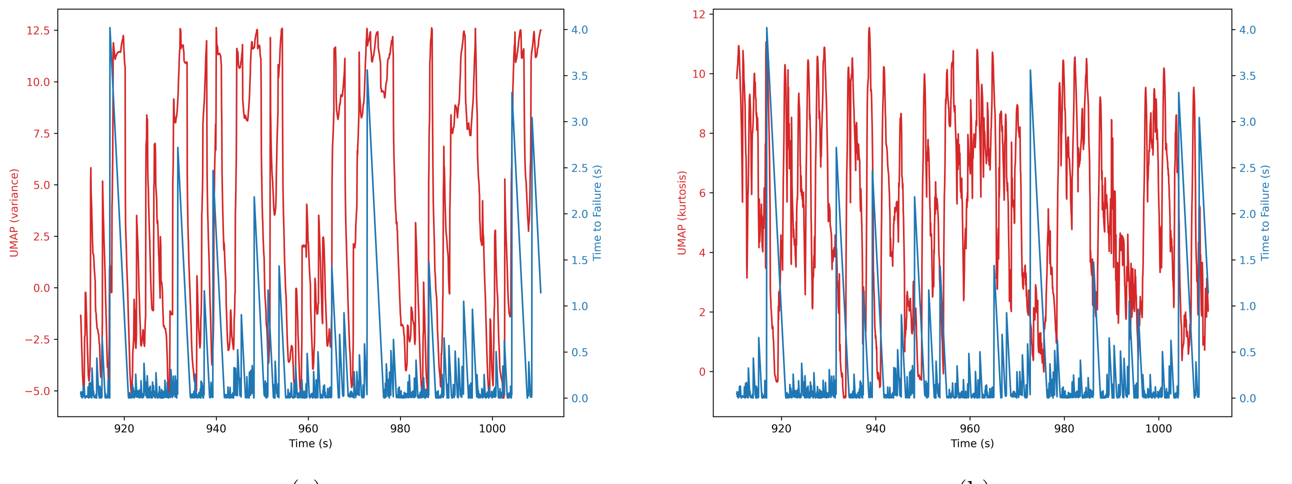
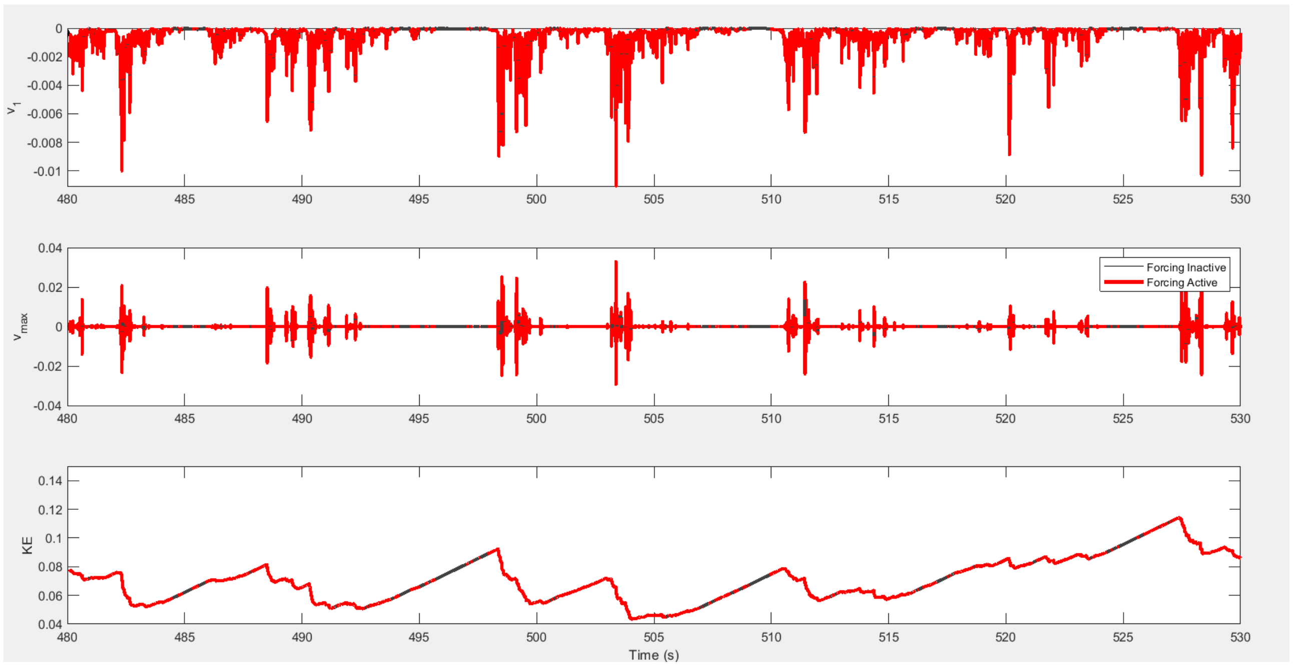


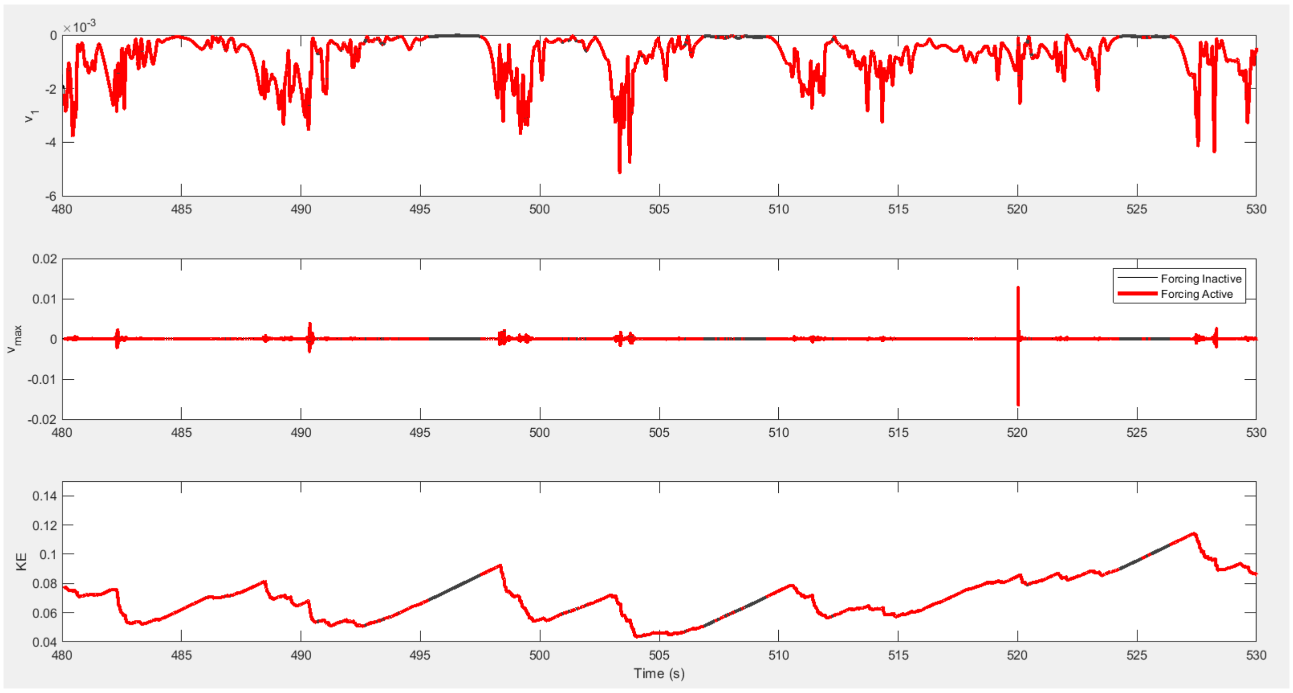
Figure 22: The first component of the UMAP analysis plotted against time, compared with the time to failure.

3.3.3 HAVOK Analysis

The HAVOK analysis on the kinetic energy, the envelope and the real and imaginary parts of acceleration of the full combined dataset all give very similar results. The time series are divided into 14 linear functions and one remaining non-linear function. Figure 23a shows the HAVOK analysis on the complete acceleration dataset. Overall, the non-linear function is high in magnitude during failures and slightly ahead of the failure (at most around 0.4 seconds). However, the non-linear term also shows significant magnitude in the middle of inter-failure periods at times, thus making these results ineffective for prediction as there are too many false positives. Using the envelope of the acceleration, only the real part of the acceleration or only the complex part of the acceleration all lead to the same result. The HAVOK analysis on individual shells shows more promising results however (figure 23b). Shells with short periods don't show significant magnitude of the non-linear function ahead of the failures, but longer periods show significant information in the non-linear function relatively far ahead ($\sim$ one second) of failures with relatively few false positives, again suggesting the medium-long periods contain the most useful information in predicting time to failure.



(a)



(b)

Figure 23: HAVOK analysis on the full acceleration dataset (a) and the acceleration dataset of shell eight (b). Shown from top to bottom are the first linear function, which contains most information, the non-linear remainder function and the kinetic energy. Forcing is considered active when the non-linear term reaches above a certain threshold.

4 Discussion

Machine learning definitely has value and potential in the prediction of non-linear systems, though it is still far from perfect. Random Forest, Temporal Convolutional Networks and Long-Short Term Memory all have strengths and weaknesses. The main problem with this data set is that due to the non-linear nature of the data, the full combined acceleration with all periods is often unable to produce a proper prediction for high time to failure values. Separation of the periods in the shells can in some cases lead to better results, and as the full combined dataset is just a summation of the individual shells, the relevant data is actually available in the full dataset. However, because this relevant data is mostly found within the medium-long periods this data gets drowned out by much higher amplitude short period data. In figure 12, the amplitude of the acceleration data from shell 14 with a lot of inter-failure high frequency information is significantly higher than the amplitude of the acceleration data from shell eight. Compared to the full combined data (figure 3b), the amplitude of shell eight is nearly negligible low during long inter-failure periods. The inter-failure information found within the shells with short periods is very similar in form to the wave patterns of small failures, possibly leading the algorithms to falsely identify these inter-failure wave patterns as failures and thus making faulty predictions.

The performance of some of the individual shells, especially shell eight, give great promise to the potential of the models to be used in real world applications. However, as this dataset is constructed as separate shells, using only specific periods as input is trivial. In real world measurements such as seismograms, separation of the periods presents a much more challenging task. There are methods of filtering out specific frequencies from time series, but none will lead to as perfect a separation as the separate shells used in this project. The RF model trained on data filtered with a low-pass filter are the best comparison to what would potentially be possible with real data. This model is not as good as predicting high time to failure as the TCN model trained on shell eight is, but it does come relatively close. The filters used are not perfect, so some remnants of the short period inter-failure information will almost always remain in the filtered data. Finding the exact frequencies that need to be filtered out is a time consuming task though, as many different combinations need to be tested to find the best frequencies to be used for prediction. While the short period data contains a lot of irrelevant information, this does not mean that there is not also important information to be found within these periods. Extracting the relevant information from these high frequencies while removing the irrelevant information is a difficult task however.

Overall the TCN and RF models perform better than the LSTM model. TCN has the highest potential when applied to the right data, but it is much more sensitive to the input data than RF. RF performs significantly better when using variance and kurtosis as input data and on acceleration data filtered through a Chebyshev filter.

4.1 Data Analysis

Data analysis tools such as PCA, UMAP and HAVOK are highly valuable in machine learning, as a big challenge is finding the right data to use as input, where these methods can be a great help. In the PCA and UMAP analysis of the turbulence data, it is clear that there is relevant data in the variance and kurtosis of acceleration for the prediction of time to failure, though there are still multiple exceptions. Models trained to predict the time to failure with just the variance and kurtosis as input do not achieve the performance as models trained on the full acceleration data, implying that there is additional significant information in the acceleration that can not be extracted by the variance and kurtosis.

HAVOK analysis shows that the time series can be described linearly during most of the inter-failure periods, while failures need a lot of contribution from non-linear terms. The non-linear term becomes significant slightly before the failure, but also during inter-failure high-frequency waveforms, preventing the use of this non-linear term for reliable prediction. HAVOK analysis of acceleration data from shell eight tells us that the non-linear term becomes significant further ahead of the failure than for the full acceleration data set and shows less influence from noise during inter-failure periods. However, the clearest HAVOK analysis still only shows the non-linear term significant around one second before the failure, while TCN on acceleration data is able to make relatively accurate predictions of an upcoming failure up to three seconds ahead of time (figure 13b). This suggests that while the HAVOK analysis can tell us more about the nature of the data, it does not fully explain what the TCN algorithm bases its prediction on.

Other data analysis must be done both on this data and real world data. It might be that acceleration data is not the best data to use for prediction, or that there is a method to better extract the most useful information from the acceleration data while discarding the rest.

4.2 Forecasting of Kinetic Energy

The forecasting of kinetic energy attempted in this project has not lead to accurate models. When these forecasting attempts are compared to the forecasting of stress in lab-quakes, we can see a potential explanation for the poor performance. Experiments done by Laurenti et al. (2022) show that while forecasting can be done with great accuracy in many cases, if the failure does not have significant creep before the drop in stress, the models are mostly unable to give accurate forecasts. In case of the simulated turbulence data, the failures are mostly very sudden as well without much pre-failure warning in the kinetic energy curve. Besides the creep, the lab-quakes are also highly periodic. If every failure follows the previous failure by a very consistent amount of time with a very small deviation, forecasting is relatively simple even without the use of machine learning techniques.

4.3 Machine Learning on Nonlinear Data

The big problem with this data is that it is chaotic and non-periodic. This is especially apparent in the forecasting of the kinetic energy, where the non-periodicity of the data results in highly inconsistent and often useless models. Forecasting models on nonlinear time series have mostly been successful if there is at least some degree of periodicity present in the data (Raissi & Karniadakis, 2018) such as in weather forecasting (Avazov & Khoussainov, 2019)e. While there is quite a lot of non-periodicity present in the weather data, the weather is of course also significantly influenced by the time of year and more locally the time of day as well, which gives the data periodicity. Machine learning models on nonlinear data has also been somewhat more successful when using statistical reinforcement (Ryo & Rillig, 2017). It might be possible to use statistical reinforcement into consideration when using machine learning on lab-quakes and real-world earthquakes, as statistics have already proven to be a valuable tool in risk assessment in those fields. However, the data of the turbulence simulation is so non-periodic, that statistical analysis likely will not lead to more accurate models. The characteristics of lab created earthquakes will almost always create periodicity. There are some experiments with machine learning on lab-quakes that have a slightly more aperiodic nature (Tanyuk et al., 2019), but even these are still much more periodic than the trubulence simulation data. Machine learning application on highly non-periodic data such as the turbulence simulation data has not been explored much yet.

4.4 Machine learning on real-world earthquakes

Application of machine learning on earthquakes has already been attempted multiple times. Machine learning has seen great success in estimating the magnitude of an earthquake even when using just a single station (Mousavi & Beroza, 2019), while previously the calculation of the magnitude of an earthquake necessitated the use of multiple stations, extensive post-processing of the data and research in how the local geology affects seismic waves. This allows early warning of the magnitude of an earthquake based on the first incoming P-waves, giving advance warning to people in the area. These techniques are able to, at best, give a warning a few minutes ahead of the arrival of damaging surface waves and mainly serve as an estimation for how bad the situation is so that proper emergency response plans can be activated.

Classification machine learning techniques are also in wide use for estimating the level of damage and casualties following an event (Mangaluthu et al., 2020; Sajan et al., 2023). This is also mainly a technique relevant for emergency response teams and is not able to predict earthquakes ahead of time.

Machine learning has also shown some levels of success in estimating the maximum magnitude of earthquakes in the next year based on the earthquakes of the previous year (Last et al., 2016). Rafiei & Adeli, 2017, succeeded in making a model that is able to predict the magnitude and location of an upcoming earthquake in a few weeks ahead of the earthquake itself, provided that there is sufficient data. While these models are very useful, these models are not able to predict when the earthquake is occurring. A model might be able to predict that an earthquake will happen in the next year, but will not be able to predict whether that earthquake will occur within one month or 12 months from the present. Rundle et al., 2022 attempted to use machine learning to forecast fault stress just one time step (one-thirteenth of a year) in the future, but the model was consistently unable to forecast large earthquakes.

While the turbulence simulation data provides great insight in the potential and workings of the several machine learning techniques, it is not directly translatable to real-world earthquakes. The turbulence data is a one dimensional isolated system, which is not representative of real faults, which are of course three dimensional. The occurrence of earthquakes might also be influenced by other nearby earthquakes (Chen et al., 2013) further complicating the forecasting and prediction. This phenomenon is also recognisable in the turbulence simulation data, where the majority of failures occur within failure sequences with little time between failures, and relatively few failures occur on their own. Seismometers will also record signals emitted by other faults and noise from wind, waves, traffic and even the seismometer itself (Ringler & Hutt, 2010). This noise will make it difficult for algorithms to identify precursor signals coming from the fault of interest. Even if the acoustic emissions from the fault of interest can be perfectly separated from all other noise, earthquake prediction will still be a highly complex task. The exact physics behind earthquakes is dependant on the surrounding material of the fault, fault movement, length and depth as well as the presence of water and magma (Zhao et al., 2002). So, if it is ever possible to create a model for prediction of earthquakes, it will most likely only be applicable for a single location along a single fault. Any other location, even among the same fault will quite possibly need to be trained on data recorded as close as possible to the location of interest. As the machine learning models require large amounts of well-labelled data, the application of machine learning techniques will most likely only be possible at locations with extensive records of seismicity and large catalogues of both small and large earthquakes. A big problem with the simulation of turbulence data is also the complete absence of creep of any kind. In lab created earthquakes, creep is almost always present in some form, giving clear advance warning of an upcoming failure.

The machine learning methods have not yet been used to predict magnitude of the failure at the same time as the time to failure, besides the attempts of forecasting kinetic energy. The magnitude of a failure event could potentially be estimated by producing accurate predictions of both the start time of the failure and the end time of the failure. The duration of the failure relates to the magnitude of the failure, with longer duration failures typically releasing more energy than short failures. This is not an entirely consistent rule, so it should only be taken as a rough estimate of the magnitude. An accurate forecasting model for kinetic energy or stress could also give valuable predictions for both the time and magnitude of an upcoming failure, though this is a much bigger challenge for non-periodic time series than just the prediction of time to failure.

5 Conclusion

Machine learning can definitely be used for non-linear time series analysis. There is not one single algorithm that consistently performs better than the others, each have their own strengths and weaknesses. The main hurdle for high accuracy prediction of especially high time to failure is the presence of high frequency acceleration data in the inter-failure period, leading the algorithms to falsely identify this data as a failure. With the turbulence simulation data it is relatively trivial to extract the periods that contain mostly useful information. This is not always achievable for non artificial acceleration data, though the use of frequency filters can improve performance. Forecasting of kinetic energy is mostly unsuccessful. Part of this is due to the non-periodic nature of the data, part of this is the lack of pre-failure creep which can often be seen in lab earthquakes. Application of machine learning on real-world earthquakes might be possible, but a lot of data analysis is needed to be able to extract the data with the most useful information as well as filtering out of noise data generated by other faults, natural and human processes. Besides that, real-world faults are highly complex systems with great variation, preventing a one-size-fits-all solution. Extensive data analysis is necessary to be able to use the machine learning techniques to their full potential and giving us a better understanding of the physics behind failures.

6 Acknowledgements

I would like to thank Prof. dr. Jeannot Trampert and Rens Elbertsen for their supervision and help with the subject. I also thank the various peers, friends and family members who provided helpful feedback and motivation.

References

Ali, M., Jones, M. W., Xie, X., & Williams, M. (2019). TimeCluster: dimension reduction applied to temporal data for visual analytics. *The Visual Computer*, 35(6-8), 1013-1026.

Avazov, N., Liu, J., & Khoussainov, B. (2019, July). Periodic neural networks for multivariate time series analysis and forecasting. *2019 international joint conference on neural networks (IJCNN)*, 1-8. IEEE.

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv*, 1803.01271.

Benzi, R., Castaldi, I., Toschi, F., & Trampert, J. (2022). Self-similar properties of avalanche statistics in a simple turbulent model. *Philosophical Transactions of the Royal Society A*, 380(2218), 20210074.

Briggs, P., Press, F., & Guberman, S. A. (1977). Pattern recognition applied to earthquake epicenters in California and Nevada. *Geological Society of America Bulletin*, 88(2), 161-173.

Brunton, S. L., Brunton, B. W., Proctor, J. L., Kaiser, E., & Kutz, J. N. (2017). Chaos as an intermittently forced linear system. *Nature communications*, 8(1), 19.

Chen, K. H., Bürgmann, R., & Nadeau, R. M. (2013). Do earthquakes talk to each other? Triggering and interaction of repeating sequences at Parkfield. *Journal of Geophysical Research: Solid Earth*, 118(1), 165-182.

Chen, H. J., Chen, C. C., Ouillon, G., & Sornette, D. (2017). Using geoelectric field skewness and kurtosis to forecast the 2016/2/6, ML 6.6 Meinong, Taiwan Earthquake. *Terrestrial, Atmospheric and Oceanic Sciences*, 28(5), 745-761.

Chollet, F. & others. (2015). Keras. `https://keras.io`

Corbi, F., Sandri, L., Bedford, J., Funiciello, F., Brizzi, S., Rosenau, M., & Lallemand, S. (2019). Machine learning can predict the timing and size of analog earthquakes. *Geophysical Research Letters*, 46(3), 1303-1311.

Deng, S., Zhang, N., Zhang, W., Chen, J., Pan, J. Z., & Chen, H. (2019, May). Knowledge-driven stock trend prediction and explanation via temporal convolutional network. *Companion Proceedings of The 2019 World Wide Web Conference*, (pp. 678-685).

Gelfand, I. M., Guberman, S. I., Izvekova, M. L., Keilis-Borok, V. I., & Ranzman, E. J. (1972). Criteria of high seismicity, determined by pattern recognition. *Tectonophysics*, 4, 415-422.

Gelfand, I. M., Guberman, S. I., Keilis-Borok, V. I., Knopoff, L., Press, F., Ranzman, E. J., Rotwain, I. M., & Sadovsky, A. M. (1976). Pattern recognition applied to strong earthquakes in California. *Physics Earth and Planetary Interiors*, 11(3), 227-283.

Geller, R. J., Jackson, D. D., Kagan, Y. Y., & Mulargia, F. (1997). Earthquakes cannot be predicted. *Science*, 275(5306), 1616-1616.

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction (Vol. 2, pp. 1-758). New York: springer.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735-1780.

Hewage, P., Behera, A., Trovati, M., Pereira, E., Ghahremani, M., Palmieri, F., & Liu, Y. (2020). Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station. *Soft Computing*, 24, 16453-16482.

Ho, T. K. (1995, August). Random decision forests. *Proceedings of 3rd international conference on document analysis and recognition*, 1, 278-282. IEEE.

Kagan, Y. Y., & Knopoff, L. (1987). Statistical short-term earthquake prediction. *Science*, 236(4808), 1563-1567.

Koopman, B. O. (1931). Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences*, 17(5), 315-318.

Lara-Benítez, P., Carranza-García, M., Luna-Romera, J. M., & Riquelme, J. C. (2020). Temporal convolutional networks applied to energy-related time series forecasting. *Applied Sciences*, 10(7), 2322.

Last, M., Rabinowitz, N., & Leonard, G. (2016). Predicting the maximum earthquake magnitude from seismic data in Israel and its neighboring countries. *PloS one*, 11(1), e0146101.

Laurenti, L., Tinti, E., Galasso, F., Franco, L., & Marone, C. (2022). Deep learning for laboratory earthquake prediction and autoregressive forecasting of fault zone stress. *Earth and Planetary Science Letters*, 598, 117825.

Lea, C., Vidal, R., Reiter, A., & Hager, G. D. (2016). Temporal convolutional networks: A unified approach to action segmentation. *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III*, 14 (pp. 47-54). Springer International Publishing.

Li, X., & Wu, X. (2015, April). Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. *2015 ieee international conference on acoustics, speech and signal processing (icassp)*, 4520-4524. IEEE.

Lin, C. H. (2009). Foreshock characteristics in Taiwan: Potential earthquake warning. *Journal of Asian Earth Sciences*, 34(5), 655-662.

Lomnitz, C. (1966). Statistical prediction of earthquakes. *Reviews of Geophysics*, 4(3), 377-393.

Mangalathu, S., Sun, H., Nweke, C. C., Yi, Z., & Burton, H. V. (2020). Classifying earthquake damage to buildings using machine learning. *Earthquake Spectra*, 36(1), 183-208.

Marzocchi, W., & Zechar, J. D. (2011). Earthquake forecasting and earthquake prediction: different approaches for obtaining the best model. *Seismological Research Letters*, 82(3), 442-448.

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv*, 1802.03426.

Mousavi, S. M., & Beroza, G. C. (2020). A machine-learning approach for earthquake magnitude estimation. *Geophysical Research Letters*, 47(1), e2019GL085976.

Nozhnitski, Y. A., Lokshtanov, E. A., Dolgopolov, I. N., Shashurin, G. V., Volkov, M. E., Tsykunov, N. V., & Ganelin, I. I. (2006, January). Probabilistic prediction of aviation engine critical parts lifetime. *Turbo Expo: Power for Land, Sea, and Air*, 42401, 1025-1034.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *The Journal of machine Learning research*, 12, 2825-2830.

Rafiei, M. H., & Adeli, H. (2017). NEEWS: A novel earthquake early warning model using neural dynamic classification and neural dynamic optimization. *Soil Dynamics and Earthquake Engineering*, 100, 417-427.

Raissi, M., & Karniadakis, G. E. (2018). Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357, 125-141.

Remy, P. (2020). Temporal Convolutional Networks for Keras. *GitHub repository.* `https://github.com/philipperemy/keras-tcn`.

Ringler, A. T., & Hutt, C. R. (2010). Self-noise models of seismic instruments. *Seismological research letters*, 81(6), 972-983.

Rouet-Leduc, B., Hulbert, C., Lubbers, N., Barros, K., Humphreys, C. J., & Johnson, P. A. (2017). Machine learning predicts laboratory earthquakes. *Geophysical Research Letters*, 44(18), 9276-9282.

Rovithakis, G. A., & Vallianatos, F. (2000). A neural network approach to the identification of electric earthquake precursors. *Physics and Chemistry of the Earth, Part A: Solid Earth and Geodesy*, 25(3), 315-319.

Rundle, J. B., Donnellan, A., Fox, G., & Crutchfield, J. P. (2022). Nowcasting earthquakes by visualizing the earthquake cycle with machine learning: A comparison of two methods. *Surveys in Geophysics*, 43(2), 483-501.

Ryo, M., & Rillig, M. C. (2017). Statistically reinforced machine learning for nonlinear patterns and variable interactions, *Ecosphere*, 8, e01976.

Sajan, K. C., Bhusal, A., Gautam, D., & Rupakhety, R. (2023). Earthquake damage and rehabilitation intervention prediction using machine learning. *Engineering Failure Analysis*, 144, 106949.

Sibly, P. (1977). The Prediction of Structural Failures (Doctoral dissertation, University of London).

Suzuki, Z. (1982). Earthquake prediction. *Annual Review of Earth and Planetary Sciences*, 10(1), 235-256.

Tanyuk, O., Davieau, D., South, C., & Engels, D. W. (2019). Machine learning predicts aperiodic laboratory earthquakes. *SMU Data Science Review*, 2(2), 11.

Syed, M. A. B., & Ahmed, I. (2023). A CNN-LSTM architecture for marine vessel track association using automatic identification system (AIS) data. *Sensors*, 23(14), 6400.

Thangaraj, A., & Wright, P. K. (1988). Drill wear sensing and failure prediction for untended machining. *Robotics and computer-integrated manufacturing*, 4(3-4), 429-435.

Wang, L. Y., Chen, P. Y., Wu, Z. L., & Bai, T. X. (2005). Characteristics of foreshock and its identification. *Acta Seismologica Sinica*, 18, 180-188.

Zhao, D., Mishra, O. P., & Sanda, R. (2002). Influence of fluids and magma on earthquakes: seismological evidence. *Physics of the Earth and Planetary Interiors*, 132(4), 249-267.