Machine Learning for Anomaly Detection in Lithography Machines

Cai, Yidan

Award date:
2021

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science
Data Mining Research Group

Machine Learning for Anomaly Detection in Lithography Machines

Master Thesis

Yidan Cai

Supervisors:

Mykola Pechenizkiy

Luca Troisi

Behnaz Pour Ebrahimi

version 4.0

Eindhoven, July 2021

Abstract

The goal of anomaly detection in this research is to forecast the performance issues of lithography machines, as unscheduled downtime can be costly. In this study, the Omnianomaly model[1] which has a backbone of Recurrent Neural Network (RNN) combined with Variational Autoencoder (VAE) is used for forecasting the occurrence of performance issues. The original Omnianomaly model incorporates additional latent space connection and planar normalization flow into the RNN-VAE based model. The experiment results show that the detection model is able to detect the degradation trends in some machines. The results also show that the decision threshold should be tuned for each machine individually due to the different anomaly score ranges of machines, while the model itself generalizes well across different machines. Incremental learning is applied to the detection model to observe if the model can be updated with new data for better performance. In this case, incremental learning does not help improve the performance of the detection model over time.

After the anomalies are detected, they are then classified into different failure modes. In the failure classification model, tree-based models are implemented for feature extraction, in which a small proportion of zernike features with the highest feature importance are selected. The neural networks then use these features for failure classification. In this classification task, using the features extracted by the tree-based models improves the results of neural networks on this small dataset. The failure detection model is expected to reduce the downtime in machines, but the online detection performance still needs to be further improved to achieve satisfying business value. The failure classification model is estimated to reduce hundreds of labor hours of diagnostics engineers per year.

Acknowledgement

This study is carried out as my graduation project at ASML in 2021. It is a difficult period as COVID-19 is still playing out and people are staying home. The coronavirus has caused many things to change, but science and research never stops.

I would like to express my gratitude to all of my supervisors, my graduation supervisor Mykola Pechenizkiy and company supervisors Luca Troisi and Behnaz Pour Ebrahimi. I would like to thank Luca Troisi and Behnaz Pour Ebrahimi for providing weekly feedback on the project and presentation and for giving me this opportunity to conduct research in the Diagnostics team. I would also like to thank Mykola Pechenizkiy for providing feedback and guidance throughout my thesis.

I would like to thank the Diagnostic team that I did the internship with. I thank everyone for providing such a nice working atmosphere. You are an amazing team and keep up!

I thank my friends and classmates whom I have spent a great time with and learnt a lot from during the master program. Life would have been colorless without you.

Last but not least, I thank my families for supporting my education and life for all these years. Thank you for always being there for me, no matter what happens. Thank you for giving this chance to travel to the other side of the world for studying.

Contents

List of Figures

List of Tables

Glossary of Terms

anomaly The data points showing abnormal behaviors of the machines that caused by failures in the machine. 2

degradation The process of the status of a part or a machine moving towards total failure. 2

failures The performance issues happen in the machine that lead to breakdown of the machine or sub-optimal operation. 2

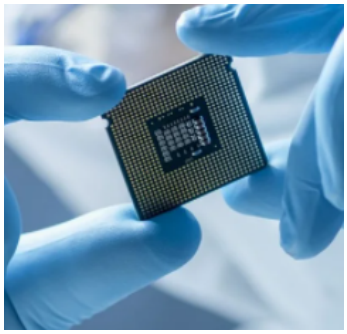
healthy Used to describe the status of the machine when the machine function normally. 2

unhealthy Used to describe the status of the machine when the machine break down, or work suboptimmally, or nearly break down ,or nearly work suboptimally. 2

Chapter 1

Introduction

ASML company provides lithography systems for the production of integrated circuits in semiconductor industry. Integrated circuit is also called microchip or chip. Chipmakers use these systems to produce integrated circuits which are important components of smartphones, cars, medical equipment and other electrical and electronic devices. Chip manufacturing is a complex process requiring hundreds of precisely controlled steps, including deposition, lithography, photoresist coating, exposure, computational lithography, baking and developing, etching, metrology and inspection, and ion implantation. Lithography machines are essential to the chip fabrication process. Thousands of signals are measured from these machines at a frequency ranging from once per day to several times per second. The failure of the parts in machines can occur anytime. The Customer Support (CS) Department is in charge of solving the failure cases of the machines of customers worldwide.



(a) A microchip

(b) A NXT 1970Ci machine

Figure 1.1: A microchip and a lithography machine

1.1 Problem Description

An unscheduled down in the lithography machine can cost chip manufacturers $20 per second. When the machines break down or not work optimally, customers report the issues and the CS Department would need to handle these issues. Due to the complexity of the systems, these issues usually need to be solved by highly experienced engineers. Diagnostics is difficult and

time consuming. Training an expert requires large investment of time and money. Even with an experienced engineer, diagnostics can take long time.

Current diagnostic system records the symptoms, the corresponding analysis and repair actions to build a knowledge database for reducing some repetitive diagnostic analysis tasks. Building this knowledge database still needs diagnostics experts so it is difficult to scale up and not sustainable in the long run. Also, lithography machines usually operate 24 hours per day, and this requires that there are always diagnostics engineers available 24 hours per day. Therefore a method to predict and diagnose abnormal behaviors automatically is desired to enable non-experts to solve most of the failure cases in a reasonable time. Forecasting of machine failure is essential to transform unscheduled down to scheduled down. A scheduled down allows more time for the preparation of maintenance such as ordering the delivery of maintained parts and therefore reduce the overall down time.

Lithography machines are complex systems and different types of failures can occur, therefore classifying the data into different failure types is also an important part of diagnostics process.

There are different types of machines and each machine can produce at least hundreds of signals at a time. Lenses are used to shape and focus light on a wafer. In order to monitor lens performances, Zernike parameters are used to measure the aberration of lenses. In this research the signals are limited to Zernike coefficients drifts and the failure modes are limited to the two most common failure types that lead to lens aberration during production. More detailed information about these different types of failures and their measurements can be found in chapter 2.

The data comes from the history signals of the machines. The repair actions are recorded but the exact time of the machines break down or start operating suboptimally remains unknown. Furthermore, there is usually delay between the start of observing anomaly and the repair action, because arranging repair actions can take time. In addition, the anomaly is mostly due to degradation so there is no clear distinction between healthy and unhealthy status. There are also occasions when machines may still function well but parts are replaced as a proactive maintenance to prevent future unscheduled down. The goal is to predict when the machines become unhealthy and need new parts replacement from the history signals and repair data.

The measured data can be affected by operations, change of parameters of the machine and environment. The model needs to take the stochasticity of the data into account.

As the model is expected to detect and predict anomalies over time, it is preferred that the model can keep training and adjusting according to new data after deployment. The model is required to take into account of the feedback from human experts. These feedback can be interpreted as new training data with labels.

Both false negative rate and false positive rate should be as low as possible. During usage, when the machine is detected as unhealthy and an alarm is raised, to confirm there are issues in the machine, the production process needs to be stopped before further examining the machine for any potential issues. As indicated before, stopping the production can be expensive for customers. Compared to a false positive, the cost of a false negative is less. Therefore a low false positive rate is preferred than a low false negative rate. The model should not raise an alarm on the anomalies unless it has enough confidence about its prediction.

It is desired that the model chosen can address as many as the above mentioned challenges as possible.

1.2 Business Value

The business value of the detection model can be calculated under certain assumption. Suppose the predictive maintenance reduces the downtime in a machine breakdown 14 hrs, and a false alarm costs unnecessary down time 24 hrs. And suppose the machine on average has a breakdown every 2 years, with 1000 machines and the model predicts $x\%$ cases. The false positive to true

positive rate is y. The total downtime saved can be calculated as the following:

$$\begin{aligned}&Total\ downtime\ saving\\&= 14hr * 0.5/machine/year * 1000\ machine * x\% -\\&y * 0.5/machine/year * x\% * 24hr * 1000\\&= x(70 - 120y)\ hr/year\end{aligned}$$

From the calculation, it can be seen that the ratio of false positive to true positive needs to be lower to a certain limit to achieve the goal of reducing downtime.

The business value of the classification model can also be calculated as the following: Suppose there are two expert rules models for the confirmation of failure modes, each take 1 hr to confirm For a random guessing model (50%), the average labor hour is

$$50\% * 1 + 50\% * 2 = 1.5\ labor\ hr$$

With a x% accuracy failure classification model, the average labor hour is

$$x\% * 1 + (1 - x\%) * 2 = 2 - x\%\ labor\ hr$$

Total labor hour saved:

$$(x\% - 0.5) labor\ hr * 1000\ machines * 0.5/year = 5x - 250\ labor\ hr/year$$

1.3 Research Questions

The research questions can be formulated as the following:

1. What kind of model and techniques can be used for anomaly detection and classification in this high-dimension, multivariate time series dataset?

2. To what extent can the model detect and predict abnormal behaviors through the Zernike coefficients drifts signals?

3. How to adjust the model according to new incoming data?

A sub research question is also posed:

How to extract useful features from this high dimension dataset?

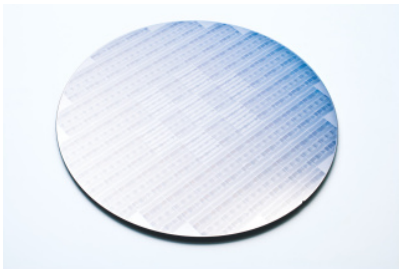
1.4 Thesis Outline

Chapter 1 describes and formulates the research questions and motivation. Chapter 2 introduces the lens problem in this case study and its relation to the data. Chapter 3 is the literature review on anomaly detection models including selected model and techniques. Chapter 4 is the exploratory data analysis. Chapter 5 contains experiment designs, settings and evaluation metrics. Chapter 6 is the results of experiments with analysis. Finally, Chapter 7 contains the conclusion and future work. .

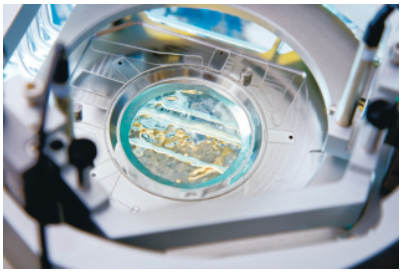
Chapter 2

Lens Aberration

A wafer is a round disk made of high purity crystalline silicon. Chips are built in a grid formation of wafer surface. Lens project the patterns on a reticle to a wafer.



A silicon wafer



Immersion system

All lens elements are susceptible to degradation effects over lifetime. These degradation includes contamination, scratches and so on. WELLE is the abbreviation of Wet Exchangeable Last Lens Element. It is the last lens element for projection lens in the immersion systems of lithography machines. WELLE is designed to protect the lens from differential cooling caused by water splashing and evaporation. It is also one of the most common failing components that lead to lens aberration. The small imperfections in the lens cause the wavefront deformation. Different parts can have different failure types. In this case, we name the two failures as failure type A and failure type B, corresponding to the two parts related to WELLE, which we call part A and part B as in Table 7.1.

Machine parts	WELLE part A	WELLE part B
Failure modes	Failure type A	Failure type B

Table 2.1: WELLE related failure types

A comparison of the light passing through a perfect lens and a lens with aberration is shown in the Figure 2.1.

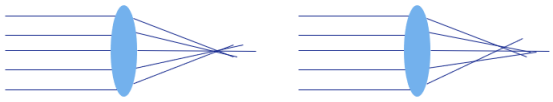


Figure 2.1: Left: perfect lens Right: lens with aberration

The wavefront is expressed as the Zernike coefficients. Zernike functions are defined in circular coordinates r, ϕ:

$$Z_n^m(r,\phi) = R_n^m(r) \begin{cases} sin(m\phi), & \text{if} \quad m < 0 \\ cos(m\phi), & \text{if} \quad m > 0 \\ 1, & \text{if} \quad m = 0 \end{cases} \tag{2.1}$$

r is the radial distance and ϕ is the azimuthal angle. n is the radial order and m is the angular frequency. Z_n^m are the Zernike coefficients. $Z_n^m(r,\phi) = R_n^m(r)sin(m\phi)$ is called odd zernike polynomials while $Z_n^m(r,\phi) = R_n^m(r)cos(m\phi)$ is called even zernike polynomials. The radial polynomials R_n^m are defined as

$$R_n^m(r) = \sum_{k=0}^{\frac{n-m}{2}} \frac{(-1)^k(n-k)!}{k!(\frac{n+m}{2}-k)!(\frac{n-m}{2}-k)!} r^{n-2k} \tag{2.2}$$

Zernike polynomials are sequences of orthogonal polynomials. There are different indices for Zernike coefficients. The one shown above uses radial order m and angular frequency n as index of Z. There are other indicing standard that use a single index, such as the OSA/ANSI indices, the Fringe/Zemax indices and the Noll's sequential indices. For the Noll's sequential indice Z_n^m is replaced by Z_j as the following:

$$j = \frac{n(n+1)}{2} + |m| + \begin{cases} 0, & m > 0 \wedge n \equiv \{0,1\} (mod\ 4) \\ 0, & m < 0 \wedge n \equiv \{2,3\} (mod\ 4) \\ 1, & m \geq 0 \wedge n \equiv \{2,3\} (mod\ 4) \\ 1, & m \leq 0 \wedge n \equiv \{0,1\} (mod\ 4) \end{cases} \tag{2.3}$$

The wavefront phase difference from the reference sphere is expressed as:

$$W(r,\phi) = \sum_n Z_n \cdot f(r,\phi)_n \tag{2.4}$$

f is the function giving the position dependence of the phase difference in a pupil plane.

The aberration expressed by different zernikes is usually visualized as the following:

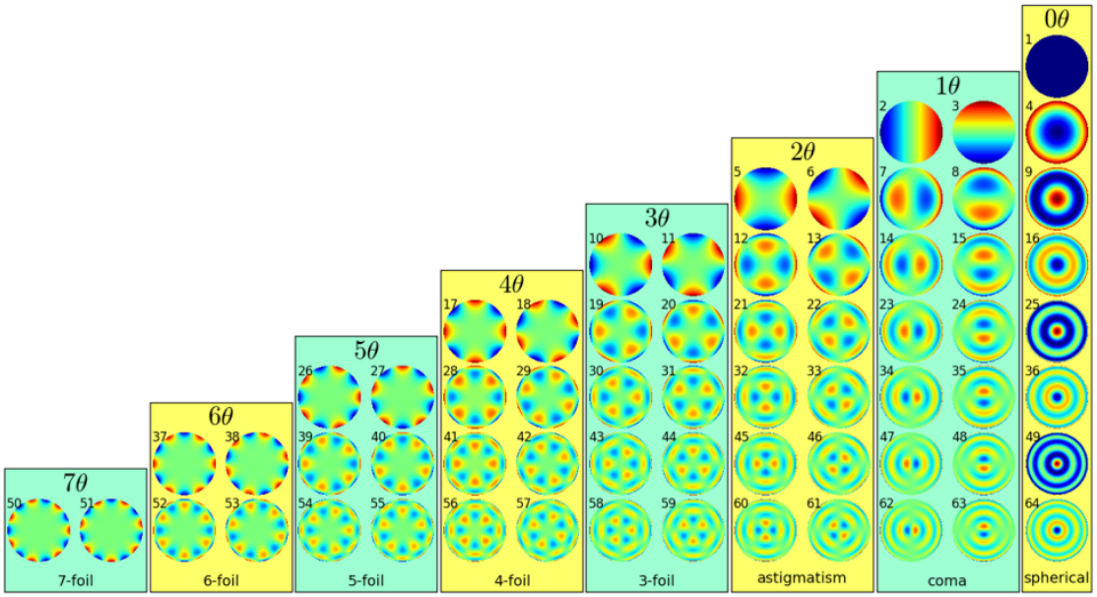


Figure 2.2: Aberration patterns expressed by Zernike polynomials(Source: ASML)

The odd zernike polynomials are related to the change of the shape and position of the aerial image. The even zernike polynomials are related to the focus of the image. The lens aberration can be adjusted by manipulating the position of lens elements. For slighter deformation of the lens elements, the system may be able to self-adjust parameters to keep the normal production process. For more severe deformation, the self-adjust mechanism is not able to offset the distortion and the production need to be stopped and the failed parts need to be replaced. Integrated Lens Interferometer At Scanner (ILIAS) is the tool that can measure and analyze wavefront aberration. It measures the optical properties of the light in the pupil plane. The lower order Zernike polynomials can be related to specific lithographic effects. Higher order zernike coefficients (above Z36) have little relations to specific lithographic effects so are typiclly not calculated [2].

The drift of each Zernike coefficient is typically measured each day. Prior the measurement, the lens system is corrected for the total drifts, Lens heating status and lens pressure. Total drift is the difference between the current value and the value at the beginning of setup. The system cannot correct all the drift. And the residual drift is the difference between the value after adjustment and the value at the beginning of the setup.

Chapter 3

Literature Review

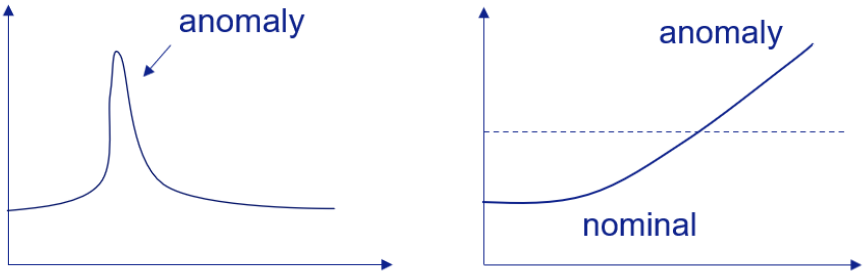
3.1 Types of Anomalies

Anomalies can be classified into three categories [3] listing as the following.

Anomaly type	Point Anomaly	Contextual Anomaly	Collective Anomaly
Description	This occurs in univariate time series. The anomaly point lies out of boundary and is considered abnormal.	A point is considered normal in one context but abnormal in another context.	The individual points are not considered abnormal themselves, but when those points showing up together then the pattern would be considered as abnormal.
Example	An unusual value of a zernike drift may indicate performance issue in the machine	Unstable zernike drifts happened just after repair action can be normal, but unstable drifts after 1 month of repair may indicate failure	One zernike drift deviated from normal value range may not indicate a failure, but multiple drifts having deviation could be a sign of failure

Table 3.1: Types of anomalies

In [4] time series anomalies are classified into amplitude and shape anomalies. To capture the shape information, the autocorrelation representation of the time series is used. There are two occasions anomaly happened. One is that a whole time series or an interval of the series can be labeled as anomaly [5]. The other is anomalies that only occurred at a data point. The first occasion is believed to be usually more challenge [5]. According to the patterns of anomaly, it can be outlier or degradation. Some literature refers anomaly detection as outlier detection. In outlier detection, the anomaly points deviate from the normal points significantly. This usually happen in credit card fraud detection or network intrusion detection. The degradation anomaly usually occurs due to the degradation or wear-out of mechanical parts in machines, or the progression of disease in healthcare. Degradation anomaly is often related to fault diagnosis and prognostics problem. The anomaly points are usually occur when there are performance issues or faults.



Outlier type anomaly Degradation type anomaly

Figure 3.1: Two different types of anomaly

As there is usually no clear line to distinguish between healthy and unhealthy status, typically in prognostics the task is to calculate the healthy index that represents the degree of healthiness and to estimate remaining useful life (RUL) as in [6][7].

3.2 Time Series Data Analysis

3.2.1 Univariate Time Series Analysis

Two common techniques for modeling univariate time series data are autoregressive (AR) [8] and moving average (MA) [9]. A k^{th} order MA process can be written as:

$$y_t = \mu + \sum_{i=0}^{k} \theta_i e_{t-i} \tag{3.1}$$

y_t is the time series data. e_t denotes the white noise and θ are the coefficients that need to be estimated. A p^{th} order AR model is defined as:

$$y_t = c + \sum_{i=1}^{p} \psi_i y_{t-i} + e_t \tag{3.2}$$

where c is a constant and ψ are the parameters of the model. An Autoregressive Moving Average(ARMA) is the combination of the two above models and is written as:

$$y_t = \alpha + \sum_{i=0}^{k} \theta_i e_{t-i} + \sum_{i=1}^{p} \psi_i y_{t-i} \tag{3.3}$$

where α is a constant. Using the above models require the time series to be stationary and differentiating is used to transforming time series data to stationary data.

3.2.2 Multivariate Time Series Analysis

If each of the time series in a multivariate time series dataset is modeled as univariate time series individually, the information related to the interaction between different time series may lost. A common way to model a multivariate time series analysis is Vector Autoregression (VAR) [10]. In VAR, the y_t in Equation 3.2 is replaced by a vector including all dimensions in the multivariate time series.

3.3 Anomaly Detection Methods

Anomaly detection has application in many domains including finance, healthcare sector, manufacturing and so on. In the case of detecting outlier anomalies, anomaly detection is also called outlier detection or novelty detection. In the case of degradation anomaly, anomaly detection is usually linked to prognostics. Anomaly detection approaches can be classified into knowledge-based approach, physics-based approach, data-driven approach and hybrid approach according to the content and method used to build the model [11]. Physics-based approach requires specific domain knowledge to build the model. The data-driven approaches that do not require specific domain knowledge includes statistic process control and machine learning methods.

3.3.1 Statistical Process Control

Statistical Process Control (SPC) is for monitoring sequential process [12]. There are different control charts including Shewhart chart [13], cumulative sum (CUSUM) chart [14], exponentially weighted moving average (EWMA) chart [15]. SPC is often divided into two phases. In phase I the process starts from the beginning until it reaches a stable state while at phase II the process is believed to be in a stable state and the goal is to keep this state runs stably [12]. This requires the line between phase I and phase II to be determined.

Statistics based anomaly detection method usually assumes the data follow a particular probability distribution and the data showing low likelihood are determined as outliers. A common way of estimating the probability of data points distribution is using Mahalanobis distance. This requires the parameters of the data such as mean and covariance metric to be estimated. Minimum Covariance Determinant(MDC) is a method for estimating the covariance matrix of the data that tries to minimize the distortion of anomalies [16]. This outlier detection method usually assumes data follow an ellipsoidal probability distribution (e.g., Gaussian distribution) but the real life data may not satisfy this condition.

SPC assumes the linearity and unimodality of the data which is a limitation [17].

3.3.2 Machine Learning Methods

Anomaly detection machine learning models can be classified into unsupervised models, semi-supervised models and supervised models. Supervised learning requires all data to be labeled. In unsupervised learning there is no labeled data for training. In semi-supervised learning only part of the healthy data is labeled and other data is unlabeled. Some machine learning models can be applied in both semi-supervised and unsupervised cases, so here these models are included in unsupervised category.

Unsupervised Machine Learning

Clustering can be used in anomaly detection such as in [18][19][20]. The two common clustering categories are density-based and distance-based algorithms [21]. Izakian and Pedrycz use fuzzy c-means(FCM) [22] clustering for anomaly detection in time series data [4]. Local outlier factor [23] is an anomaly detection method that applies k-nearest neighbor and determines the degree of isolation at its location based on the surrounding neighbourhood.

Principle Component Analysis (PCA) is a dimension reduction technique. It can compress the data to a lower dimension and reconstruct data and the reconstruction error can be used for anomaly detection as in [24] [25].

One-class support vector machine (SVM) [26] is used for semi-supervised anomaly detection. It estimates a function that is positive applying on a subset but negative on its complement [26]. One-class SVM is sensitive to outliers thus it is less suitable for unsupervised anomaly detection. Robust one-class SVM [27] and Eta one-class SVM [28] are proposed to reduce the effect of outliers, which can be the anomalies in the training data, and make the one-class SVM more suitable for unsupervised anomaly detection.

Deep learning is also used for semi-supervised anomaly detection. One method is using autoencoder to learn the data representation of normal data and measure the reconstruction loss to find outliers. The method is formally defined in [29] as the following:

$$\underset{A,B}{\operatorname{argmin}} \sum_{t=1}^{n} \Delta(B(A(x_t)), x_t) \tag{3.4}$$

Here A denotes the encoder $A : \mathbb{R}^m \to \mathbb{R}^p$ and B denotes the decoder $B : \mathbb{R}^p \to \mathbb{R}^m$ where $p < m$. Δ is the reconstruction loss calculation function. x_t denotes input sample. This method assumes that the autoencoder can reconstruct the normal data with low loss but is unable to reconstruct the abnormal data with low loss. However, this assumption is not necessarily valid as the autoencoder may be able to reconstruct the abnormal data as well. MemAE [30] is proposed to address this problem. In MemAE the network keeps a pool of normal data representation during training. During testing, the decoder would select a most similar representation from this pool to decode. As the reconstructed data tends to be close to normal data, the reconstruction loss of the abnormal data is expected to be higher. Sakurada and Yairi demonstrated that autoencoder can detect subtle anomalies where linear PCA fails [31]. The authors also found using a denoising autoencoder yields higher accuracy [31]. Senanayaka et al. [32] apply VAE and RNN for prognosis of bearing life and show that the model detect the upward trends of health indicator in bearings. Another deep learning method for anomaly detection is time series prediction thus is only applicable for time series data. It uses prediction loss as an indication of outliers. Both reconstruction and prediction methods need a threshold to be determined to classify normal data and outliers. The threshold can be selected to achieve a balance between precision and recall.

The semi-supervised method, or a one-class classifier often works under the assumption that the model output of anomalies differ from the model output of normal data. The semi-supervised model has the capacity to detect unseen anomalies. Kawachi et al. proposed a supervised VAE by introducing a different probabilistic representation for the anomalous data to improve the ability of detecting seen anomalies[33].

Many of the above mentioned machine learning anomaly detection methods do not specifically take into account the time series nature of the data.

Different machine learning techniques can be ensembled to achieve better performance. For example, Erfani et al. use a deep brief network to extract features then use a one class SVM with linear kernel for anomaly detection to achieve computational efficiency with comparable accuracy [34].

Autoencoder

A basic autoencoder can be viewed as an unsupervised machine learning method. Autoencoder was first proposed by Ruineihart et al. as a way of representation learning [35]. A sketch of the structure of an autoencoder can be seen in Figure 3.2. The encoder compresses the data to a lower dimension and the decoder decompress to reconstruct the data. If we denote the encoder transformation as $f(x)$ and the decoder transformation as $g(x)$, the input data is x, then the reconstruction loss is defined as $|x - g(f(x))|$. Using autoencoder for anomaly detection usually assumes that the trained autoencoder can reconstruct normal data but performs poorly when reconstructing abnormal data. This assumption may not necessarily hold, as there is no rule to limit the capability of the model to reconstruct abnormal data. A regularization term can be added to the loss function to reduce the overfitting [36].

There are many variants of autoencoders such as denoising autoencoder, sparse autoencoder, deep autoencoder, contractive autoencoder and variational autoencoder. Contractive autoencoder adds the squred Frobenius norm of the Jacobian as a regularizor therefore the loss function can be written as in [37]:

$$\mathcal{L} = \mathcal{L}(x, g(f(x))) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

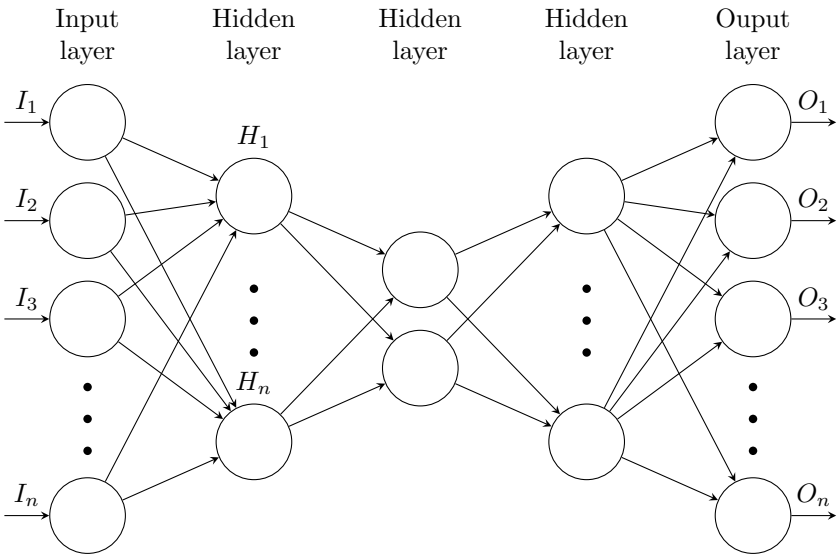


Figure 3.2: A typical autoencoder structure

A k-sparse autoencoder does not use the structure of reducing nodes in the hidden layers but instead add constrain to the activation function. In each feedforward process only the largest k activation are kept and the rest are set to zero [38].

Supervised Machine Learning

In supervised learning, the data are labeled so the anomaly detection problem could be viewed as an binary classification in an unbalanced dataset. There are different oversampling methods to achieve a more balanced dataset such as SMOTEBoost [39]. Linear discriminate analysis [40] is a dimension reduction technique to construct features that maximize the difference between different classes. Logistic regression [41] uses logistic function to model the binary variables. The limitation of logistic regression is that it assumes data are linearly separable. Decision tree can deal with non-linear boundary between classes and the model has the property to handle missing data. Decision tree [42] also has higher explainability as the decision making process is broken down into each individual decision [43]. Random forest ensembles multiple individual decision trees and each decision tree is trained on a subset of the training dataset. Gradient Boosting Machine [44] uses boosting paradigm and produces competitive result for both regression and classification. Extreme gradient boosting machine [45] uses more regularized model to control overfitting and also enhances computation efficiency. LightGBM uses Gradient-based One-Side Sampling and Exclusive Feature Bundling to further improve the efficiency [46] of gradient boosting. Deep learning can also be used in supervised anomaly detection for building a classification model.

3.3.3 Prognostics for degradation anomaly

Prognostics are commonly used for fault prognosis and remaining life estimation in mechanical parts. The degradation trend is usually non-linear, so non-linear functions such as exponential functions can be used to capture the degradation trend. In [47] the sum of two exponential functions are used to fit the degradation trend. Jagath et al. apply Autoencoder for feature extraction then use RNN for predicting healthy index of bearings with another RNN to predict RUL from the healthy index [7].

3.4 OmniAnomaly Model review

Su et al. proposed a stochastic Recurrent Neural Network model for multivariate time series anomaly detection [1]. The model achieves better and more stable performance across different datasets on four real-life datasets compared with other LSTM based autoencoder models, namely LSTM-NDT [48], DAGMM [49] and LSTM-VAE [50]. The evaluation metrics are precision, recall and F1 score. OmniAnomaly also showed satisfying performance evaluating on hitrate on the selected datasets. Following the techniques related to this model are reviewed. The researchers of OmniAnomaly have demonstrated effectiveness of these applied techniques through different experiments.

3.4.1 Model-relevant Techniques Review

Gated Recurrent Unit

Both Gated Recurrent Unit(GRU) [51] and Long Short-Term Memory(LSTM) [52] are variants of Recurrent Neural Networks [53]. The idea of RNN can be expressed using a recursive function:

$$h_t = f(h_{t-1}, x_t) \tag{3.5}$$

where h_t is the hidden state of the neural network at timestep t and x_t is the input at timestep t. $f(x)$ is the transformation of neural networks including the linear transformation of weights and bias and activation function. If unroll the $f(x)$ to weights W, bias b and activation function σ it can be expressed as:

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b) \tag{3.6}$$

Cho et al. combine RNN and autoencoder structure for machine translation and show that the model learns meaningful latent space representation of words and phrase [54]. Luo et al. introduce a similarity score in a stacked RNN combined with sparse autoencoder and demonstrate its capacity of anomaly detection in surveillance videos [55]. The property of memorizing the past information makes RNN useful for time series data analysis but it suffers from vanishing gradient problem [56]. Both GRU and LSTM address the vanishing gradient problem of RNN. GRU can be expressed as in [51] and [57]:

$$\begin{aligned} h_t &= \Gamma_u \widetilde{h_t} + (1 - \Gamma_u) h_{t-1} \\ \widetilde{h_t} &= f(\Gamma_r h_{t-1}, x_t) \\ \Gamma_u &= f_u(h_{t-1}, x_t) \\ \Gamma_r &= f_r(h_{t-1}, x_t) \end{aligned} \tag{3.7}$$

Here f_u is the update gate and f_r is the reset gate, both are the output from neural networks layers. $\widetilde{h_t}$ is the intermediate hidden state. LSTM [52] can be expressed as the following:

$$\begin{aligned} h_t &= \Gamma_u \widetilde{h_t} + \Gamma_f h_{t-1} \\ \widetilde{h_t} &= f(a_{t-1}, x_t) \\ a_t &= \Gamma_o h_t \\ \Gamma_o &= f_o(a_{t-1}, x_t) \\ \Gamma_u &= f_u(a_{t-1}, x_t) \\ \Gamma_f &= f_f(a - t - 1, x_t) \end{aligned} \tag{3.8}$$

where f_f, f_u and f_o are forget gate, update gate and output gate respectively. An additional activation state a_t is introduced.

Variational Autoencoder

Unlike autoencoder which is a deterministic model used for reconstructing the input, variational autoencoder(VAE) [58] is a probabilistic model and the output can be the likelihood of reconstructing the input. Let x be the input data and z be the latent variable. As explained in [58], VAE learns a joint distribution $p_\theta(x, z)$ that can be factorized as $p_\theta(x, z) = p_\theta(z)p_\theta(x|z)$. In autoencoder $q_\phi(z|x)$ is the encoder and $p_\theta(x|z)$ is the decoder. $p_\theta(z)$ is the prior distribution of the latent variable z. The encoder and decoder are modeled by neural networks. ϕ and θ denote the parameters of encoder and decoders. The objective function of VAE is the variational lowerbound of the marginal likelihood of the data [59]. The marginal likelihood is calculated over the marginal likelihood of all data.

$$p_\theta(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} p_\theta(x_i) \tag{3.9}$$

The log likelihood $p(x)$ can be derived as the following:

$$\begin{aligned}
\log p_\theta(x) &= \int q_\phi(z) \log p_\theta(x)\, dz \\
&= \int q_\phi(z) \log \frac{p_\theta(x, z)}{p_\theta(z|x)}\, dz \\
&= \int q_\phi(z) \log \frac{p_\theta(x, z) q_\phi(z)}{p_\theta(z|x) q_\phi(z)}\, dz \\
&= \int q_\phi(z) \log \frac{p_\theta(x, z)}{q_\phi(z)}\, dz + \int q_\phi(z) \log \frac{q_\phi(z)}{p_\theta(z|x)}\, dz \\
&= \mathbb{E}_{q_\phi(z)}[\log \frac{p_\theta(z, x)}{q_\phi(z)}] + D_{KL}(q_\phi(z)||p_\theta(z|x))
\end{aligned} \tag{3.10}$$

The second term is the Kullback-Leibler(KL) divergence between $q_\phi(z)$ and $p_\theta(z|x)$. The first term is the variational lower bound, also called Evidence Lower Bound(ELBO). KL divergence is non-negative so the $\log(x)$ is always larger than the ELBO. The problem of approximate the intractable $\log(x)$ then is transferred to maximize the first term and it can be further derived as:

$$\begin{aligned}
\mathcal{L}(q_\phi(z)) &= \int q_\phi(z) \log \frac{p_\theta(x, z)}{q_\phi(z)}\, dz \\
&= \int q_\phi(z) \log \frac{p_\theta(x|z) p_\theta(z)}{q_\phi(z)}\, dz \\
&= \int q_\phi(z) \log p_\theta(x|z)\, dz + \int q_\phi(z) \log \frac{p_\theta(z)}{q_\phi(z)}\, dz \\
&= \mathbb{E}_{q_\phi(z)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z)||p_\theta(z))
\end{aligned} \tag{3.11}$$

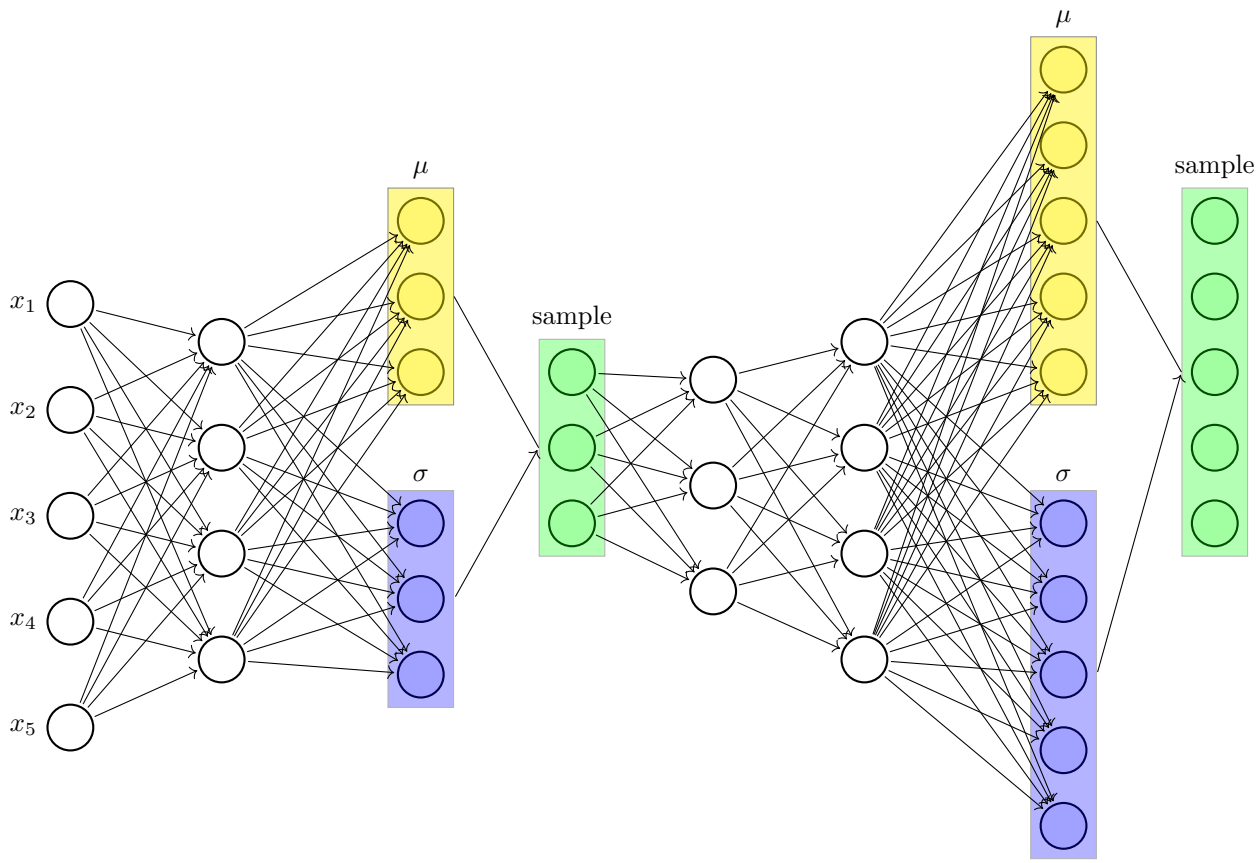


Figure 3.3: A visualization of a variational autoencoder structure

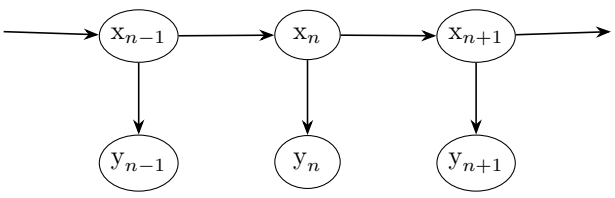
Gregor et al. developed Temporal Difference VAE which learns from separated time points of data and make prediction several time steps ahead [60].

Stochastic Variable Connection

The authors use linear Gaussian State Space Model [61] to model the connection between stochastic latent variables. As defined in [62], a state space model is consist of two time series processes: $X_t \in \mathcal{X}$ and $Y_t \in \mathcal{Y}$. As in [61], a generic state space model is given by:

$$\begin{aligned} x_n &= F_n x_{n-1} + G_n w_n \\ y_n &= H_n x_n + \epsilon_n \end{aligned} \tag{3.12}$$

Here x_n is the state and y_n is the time series observation. w_n is the input process and ϵ_n is the observation noise. The first equation in 3.12 is the transition process equation and the second equation is the observation equation. Fn is the state transition function at time n. A simplified graph indicates the state space model can be shown as:



A linear Gaussian State Space Model is given by adding an additional term as in [62]:

$$\begin{bmatrix} w_n \\ \epsilon_n \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} Q_n & 0 \\ 0 & R_n \end{bmatrix}\right) \tag{3.13}$$

Planar Normalization Flow

Planar Normalization Flow(NF) [63] is used to approximate complex posterior distribution for variational inference. Through a normalization flow, a simple initial distribution is transformed to a complex one through a sequence of invertible mappings. Let f be an invertible function, z be a random variable with distribution $q(z)$. Mapping z to a random variable using f results in $z' = f(z)$. As shown in [63], the distribution of z' is

$$q(z') = q(z)\left|det\frac{\partial f^{-1}}{\partial z'}\right| = q(z)\left|det\frac{\partial f}{\partial z}\right|^{-1} \tag{3.14}$$

and by sequentially applying the mappings K times the distribution $q_K(z)$ is obtained:

$$\begin{aligned} z_K &= f_K \circ f_{K-1} \cdots \circ f_2 \circ f_1(z_0), \quad z_0 \sim q_0(z_0) \\ q_K(z_K) &= q_0(z_0)\prod_{n=1}^{K}\left|det\frac{\partial f_n}{\partial z_{n-1}}\right|^{-1} \end{aligned} \tag{3.15}$$

3.4.2 Overall Network Architecture

The overall network architecture can be shown in Figure 3.4. Dotted arrows indicate lag transmit or non-directed transmit of the variable. The model can be viewed as two parts, an encoder generates the distribution of latent variables and a decoder generates the distribution of the reconstructed x.

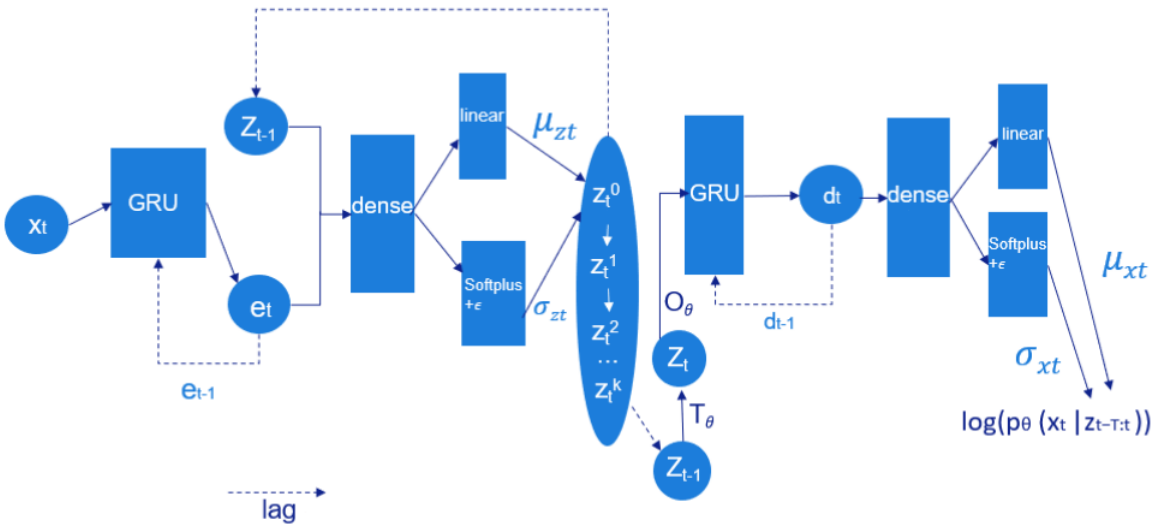


Figure 3.4: OmniAnomaly model architecture [1]

3.4.3 Limitation of semi-supervised model

The semi-supervised model is trained only on normal data and the detection is under the assumption that the trained model is less likely to reconstruct abnormal data. This assumption is not necessarily valid. There is some research that aims at resolving this problem such as Memory-Augmented VAE (MemAE) [30].

3.5 Online Learning

Online learning is also called incremental learning or continual learning. Traditionally a machine learning model is trained offline, which means all the training data can be accessed at every training time. Usually these data are fed as batches and iterating during training. When the coming data is streaming, the traditional training method may not be applicable and online learning [64] is needed. Online learning allows a model to keep training during deployment, so the model can be adjusted to new incoming data. Hoi et al. use online learning algorithm that updates the model at every incoming instance [65].

The simple approach to apply online learning on deep learning is to apply backpropagation on a single instance in each iteration but this approach has some drawbacks [66]. The model structure (e.g., number of layers) needs to be fixed before online training and this can lead to problem if the initial model is too simple or too complex. Sahoo et al. proposed a novel Hedge Backpropagation method for effectively online learning in deep neural networks [66]. In the Hedge Backpropagation, the output of neural network is:

$$F(x) = \sum_{l=0}^{L} \alpha^{(l)} f(x)^{(l)} \tag{3.16}$$

where x is the input, $f(x)^{(l)}$ is the classifier using the output at hidden layer $h^{(l)}$ and α is parameter needs to be learned. L is the number of hidden layers so the output is the weighted sum of the output of all layers. The goal of this method is to allow flexible number of layers in online learning.

3.6 Learning with small dataset

Machine learning typically needs large amount of data for good performance. In a biospectroscopy experiment, Beleites et al. show the learning curve is completely masked by random uncertainty with independent sample size 5-25 per class [67]. Pasupa and Sunhem compare shallow structure model and deep structure models on a small dataset for image classification problem and results show that deep structure models perform worse than shallow structure model when regularization and dropout are not applied. When regularization and dropout are used, the results of deep structure model and shallow structure models are comparable [68]. Another research shows that deep neural networks has lower testing accuracy than shallow structure model but a pre-trained deep neural network can perform better than shallow structure model [69]. To tackle the problem of small dataset, MacAllister et al. use meta-model, specifically Kriging and Radial Basis Functions to generate data for training a Baysian network and use Particle Swarm Optimization (PSO) to tune parameters of Baysian networks [70]. Learning from small amount of data is also known as few-shot learning which requires generalizing from a few samples. In terms of classification, usually one considers a K-way-N-shot classification in which the dataset contains k classes, each with N samples [71].

Meta-learning can be used to addressed the few-shot problem. Suppose a machine learning model has a set of parameters θ that needs to be optimized and is called a learner. A meta-learner is another model that learns to optimize θ for the learner.

3.7 Main Idea

From the above literature review, it can be seen that RNN combined with VAE takes into consideration the unbalance and time series nature of the anomaly detection problem in concern. The OmniAnomaly model adds additional techniques to enhance the performance of the model. Online learning could be used for incorporating the user feedback which can then be transformed to new training data, as online learning aims at updating the model with new training data.

Chapter 4

Data Description

All machines of this study are from TWINSCAN NXT lithography systems which use Deep ultraviolet (DUV) technology. The wavelengths used for DUV lithography are 248 nm or 193 nm. There are total 4 subclasses of machines used in this study: NXT 1950i, NXT 1960Bi, NXT 1970Ci and NXT 1980Di. The data contains the Zernike drifts, machine name, time of measurement and WELLE repair date. The Zernike drifts contain 234 dimensions. The measured zernike coefficients for the zernike polynomials ranges from 2 to 37 and each have up to 3rd order measured. Each order has up to 2 measurements, namely, Total drift and Residual Drift. In total there are 144 Residual drift measurements and 90 Total drift measurement. The Residual drift is the remaining drift after system correction.

$$Residual\ Drift = Total\ Drift - System\ corrections$$

Based on expert domain knowledge, out of the 36 zernike coefficient, 8 of them are considered as more significant indicators of type B issue and they have 64 variables. 7 zernike coefficients are considered as more significant indicators of type A issue and they are related to 46 variables. These indicators are selected by domain experts and experts have different opinions over the importance of different zernike coefficient drifts. The measurement is advised to be done once per day but based from the data collected, many machines have gaps in between measurements. Sometimes multiple measurements are done in one day. An example of some drifts recorded in a machine can be seen as the following Figure 4.3. Only 6 significant indicators are plot here for better visualization. The vertical dotted lines indicate the repair actions related to WELLE.

Figure 4.1: A machine with some Zernike drifts over recording period

Different machines show different patterns on the same zernike drifts. Below shows the same set of zernike variables on multiple machines. From the observation on these machines, the zernikes drifts gradually increase after the repair action.

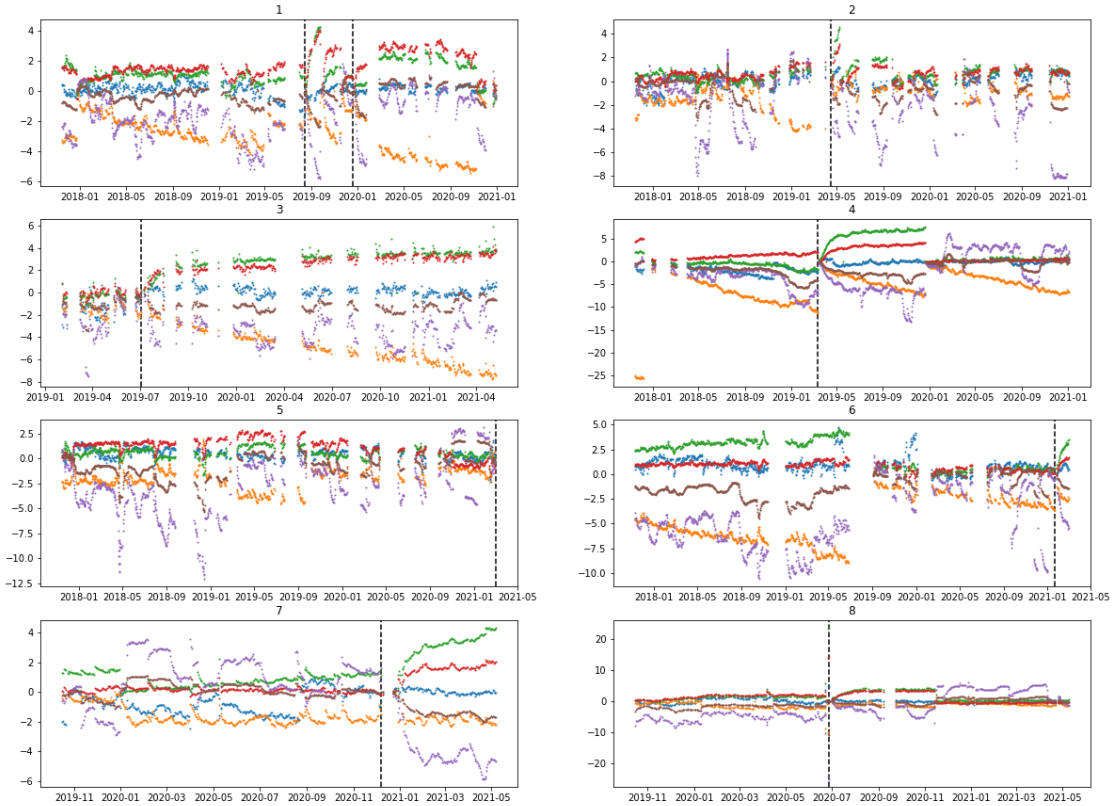


Figure 4.2: A subset of zernike drifts on multiple machines overtime

The Pearson correlation matrix calculated from the first 64 zernike variables out of the 234 zernike variables in the dataset is shown at the following. It can be seen that there are clear correlations among some of these variables. For example, usually the total drift and the residual drift of the same zernike variable have relevant high positive correlation. It could be possible to use linear dimensionality reduction techniques to reduce the dimension of this dataset.

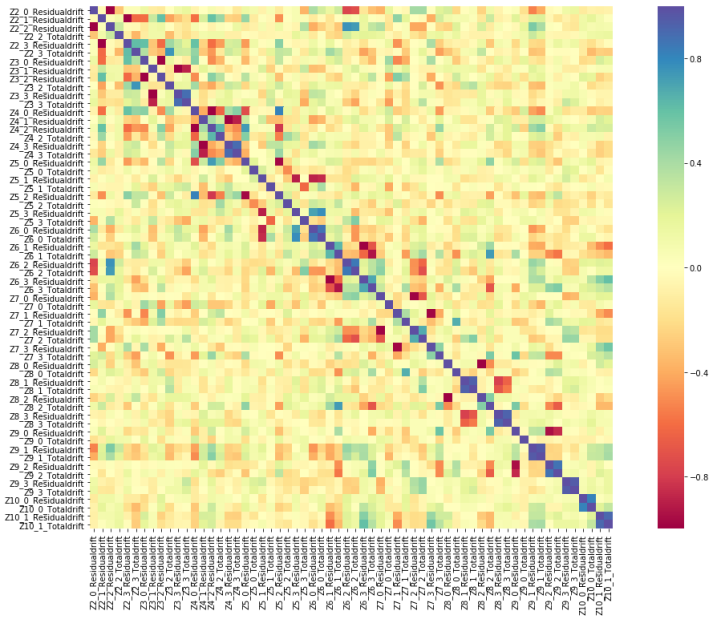
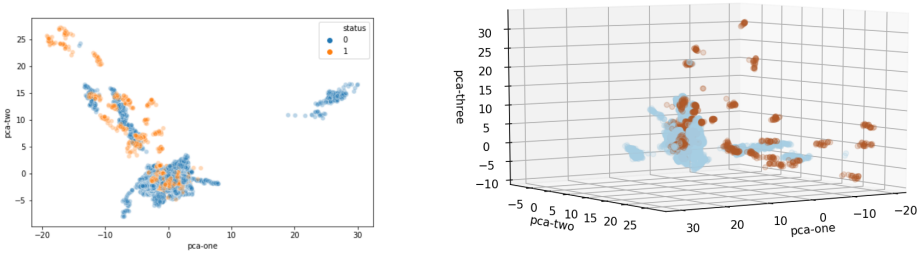


Figure 4.3: Correlation matrix plot of first 64 dimensions

Principle components analysis can be used to reduce data dimension and visualize whether data are linearly separable or not. Figure 4.4 shows the first 2 principle components and first 3 principle components from the 234 zernike variables of data points. The two different classes are healthy and unhealthy. Using Principle Component Analysis, some anomaly points can be distinguished as they scatter outside the largest cluster of normal data. There are also some healthy data deviating from the largest cluster of healthy data.

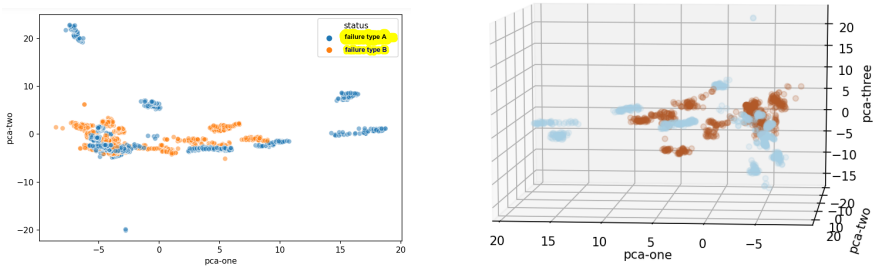


First 2 PCA components

First 3 PCA components

Figure 4.4: PCA components visualization of healthy and unhealthy data (blue: healthy, red: unhealthy)

The repair actions can be divided into proactive and reactive actions. Proactive action means to replace a part when the part is still functioning and the purpose is to prevent a future unscheduled downtime, while reactive action is to replace the part after it breaks down. In the failure classification dataset, there are around 40 machines that have reactive repair actions and will be used for the failure classification model. Figure 4.5 shows the first 2 principle components and first 3 principle components of these unhealthy data points. The two classes are type A and type B issues.



First 2 PCA components First 3 PCA components

Figure 4.5: PCA components visualization of unhealthy data

The signal data is recommended to be measured per day, but in reality this suggestion is sometimes not followed. Below Figure 4.6 shows the sampling time in one of the machine. Most of the time, the measurements are carried at the same time every day, giving two consecutive measurement a 24-hour interval. However, there are some missing measurements over time. There are also sometimes multiple measurements happen in one day.

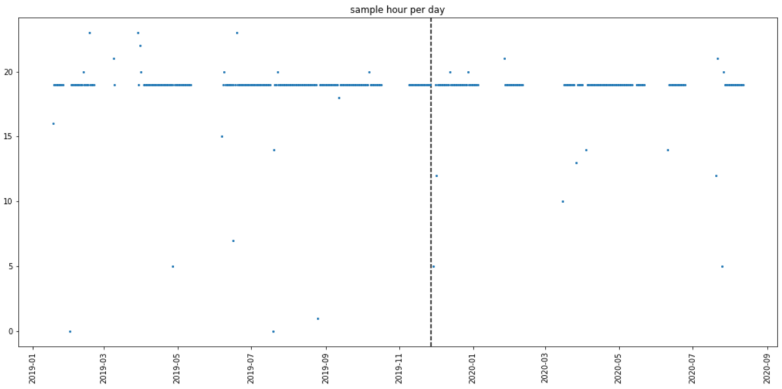


Figure 4.6: Sampling hour per day of one machine

All zernike drifts are scaled to 0-1 as a data pre-process for the machine learning model.

Chapter 5

Model Settings

There are two separate models to address the prediction and classification tasks respectively. One semi-supervised model for general abnormal behavior prediction and the other supervised model for failure classification. Notice the output of the prediction model is the opposite of reconstruction logarithm likelihood. The likelihood can be any positive value. And the logarithm likelihood can be any real number. The two failure classification categories are failure type A and failure type B. The two models are shown in the following. The detailed architecture of the OmniAnomaly model is shown in Chapter 3.

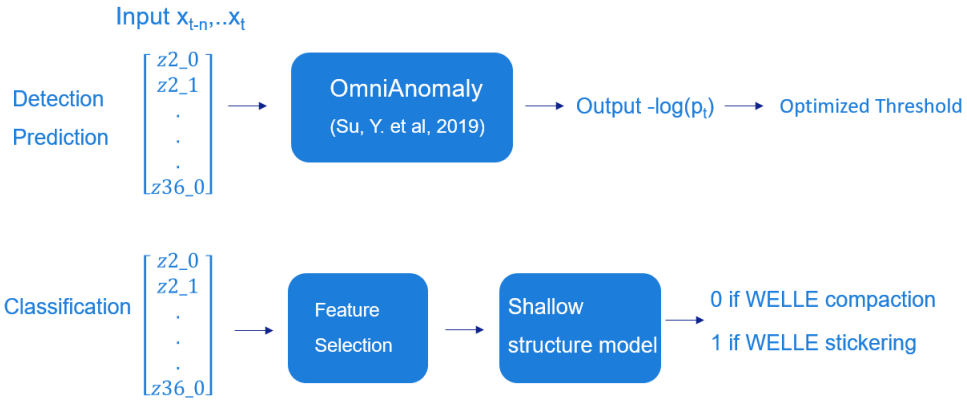


Figure 5.1: Anomaly detection and classification models overview

5.1 Anomaly Detection Model

The evaluation metrics of the detection model include the confusion matrix, precision and recall. Recall is also known as sensitivity. As indicated in the introduction, the model should avoid false positive as much as possible while maintaining a certain level of true positive. The loss of failing to predict a future failure is relatively lower, therefore a lower false positive rate is preferred to a lower false negative rate in this case. One evaluation criteria is set as "the highest recall when precision reaches 90%". The 90% precision is considered as acceptable in this business case. We would like to know how many failure cases the model can predict with the criteria that more than 90% of the cases of the model predicted are correct. Precision is calculated as:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{5.1}$$

and recall(sensitivity) is calculated as:

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{5.2}$$

When the threshold increases, precision increases, recall decreases and vice versa. The evaluation metrics can be divided into two types: one based on individual data point level and the other based on machine/case level. On the data points level, every point in every machine that belongs to testing data is counted. This can give insights about how many failure points the model can detected without compromising its confidence on the failure prediction. When calculating on the machine level, the 4 metrics in the confusion metrics are binary numbers. For example, if a machine has some false positive points after a repair action, then the false positive is counted as one, no matter how many false positive points are there. If there is no false positive then the false positive is counted as 0. This metric gives insights on the proportion that occurred failures are predicted while still satisfies the criteria that more than 90% of the time the prediction is correct. On machine level there is an additional metric which is the number of machines that are correctly classified in terms of detection. A machine is considered correctly classified only if its true positive number is larger than 0 and has no false positive nor false negative points. The thresholds are selected based on these metrics and are searched through brute force methods. Notice the optimized thresholds may be different when the threshold is optimized towards different evaluation metrics.

Metrics that do not need a threshold to be determined dynamically are also considered such as area under the receiver operating characteristic curve (ROC curve) and area under Precision-recall curve. The value of a full AUC-ROC is between 0 and 1 with 1 means perfect separation capability. As in this case, a low false positive rate is more desirable than a low false negative rate, a partial AUC-ROC with the max false positive rate set to 10% is used. The partial AUC-ROC is standardized to 1. The partial area under the precision-recall curve is calculated with the minimum precision set to 90%.

In practice, to avoid the result of one machine corrupting the overall result, the actions are also evaluated individually. Each action is labeled as either True positive, False Positive or False negative. The evaluation metrics based on these are called quality and hit rate. The calculation are as follows. Notice here the True Positive, False Positive and False Negative are based on per action not per data point. In this calculation, the whole time series of each individual machine needs to be considered and some human decisions are needed when a machine is classified as false positive, true positive, false negative or not counted into final result.

$$Hitrate = \frac{True\ Positive\ Case + False\ Positive\ Case}{True\ Positive\ Case + False\ Negative\ Case} \tag{5.3}$$

$$Quality = \frac{True\ Positive\ Case}{True\ Positive\ Case + False\ Positive\ Case} \tag{5.4}$$

The offline detection model is trained and tested using two types of training approaches. First, the partition are based on machines and cross validation is applied. At each validation, the model is trained on some machines and tested on other machines. The second experiment is "run-to-failure" which trained and tested on the same machine. Below is an illustration of the two different training approaches.

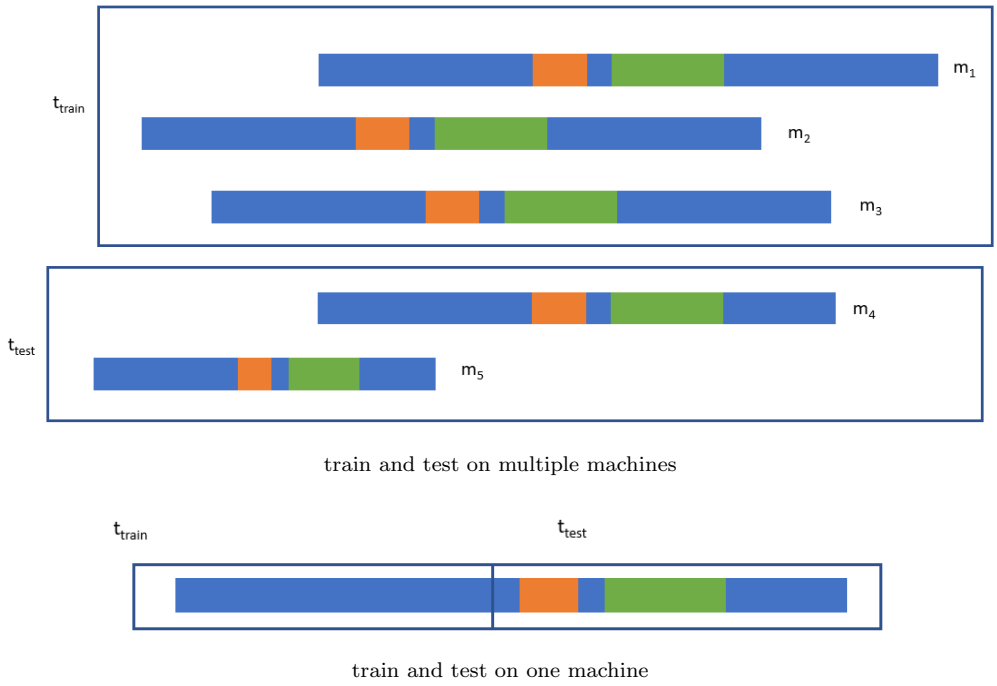


Figure 5.2: Two training and testing approaches, each multi-colored bar denotes the time series data of one machine, green denotes healthy data, red denotes unhealthy data, and blue denotes unknown

5.2 Anomaly Classification Model

The classification models are shallow structured models that are benchmarked for the failure classification task. These models include Decision tree, Extreme Gradient Boosting Machine (XGBoost), Light Gradient Boosting Machine (LightGBM), logistics regression, support vector machines and shallow neural networks. The neural networks are also trained on the important features extracted by decision tree and XGBoost. The splitting rule of decision tree here is Gini [72]. Gini impurity is calculated as:

$$Gini\ impurity = \sum p(i)(1 - p(i)) \tag{5.5}$$

where p(i) denoted the probability of samples in class i. The feature importance is calculated using the scikit-learn package in python. For decision tree, the feature importance is calculated as the decrease of Gini impurity at a node weighted by the probability of reaching that node as in [72]:

$$\frac{n_{node}}{n}(I - \frac{n_l}{n_{node}}I_l - \frac{n_r}{n_{node}}I_r) \tag{5.6}$$

where n_{node} denotes the number of samples at a node, n_l denotes the number of samples at the left children node and n_r denotes the number of samples at the right children node. n denotes the total number of samples. I denotes Gini impurity. For XGBoost, the importance is calculated by number of times of a feature appear in the constructed trees.

The proposed method that uses tree-based model as a feature selector can be viewed as in Figure 5.3:

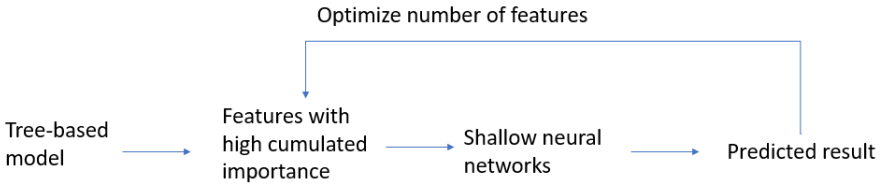


Figure 5.3: A proposed classification method

In the system design, the classification will only be triggered when the data is detected as anomaly by the detection model. The classification then classifies the potential failure into 2 failure classes. Based on history data, the frequencies of the 2 failure modes are similar. The evaluation of the classification model is accuracy, which is calculated as following:

$$Accuracy = \frac{correctly\ classified\ case}{Total\ case} \tag{5.7}$$

5.3 Online learning with detection model

The online learning model aims at adjusting the model on the new data based on the existing model which has been trained offline.

There is no actual real-time data stream for the online training settings. So the real-time data would be simulated using existing training data. The total dataset is divided into offline training set, online training set. At the beginning the model is trained on the offline training set, after that, a sliding training interval would slide across the timeline and train the model using the healthy data in every following interval. The healthy data for training is defined as within one year after the repair action and no repair action in the following sliding training interval.

The online learning simulation is applied on the first model for anomaly detection. Buffer interval days are the amount of days that are assumed as not healthy before an abnormal behavior occur. Setting a training sliding interval instead of updating the model after every training data come in so the model would not be affected by a single noisy point thus may help stabilize the training. Larger sliding period would lower updating frequency and increasing memory needed for computation.

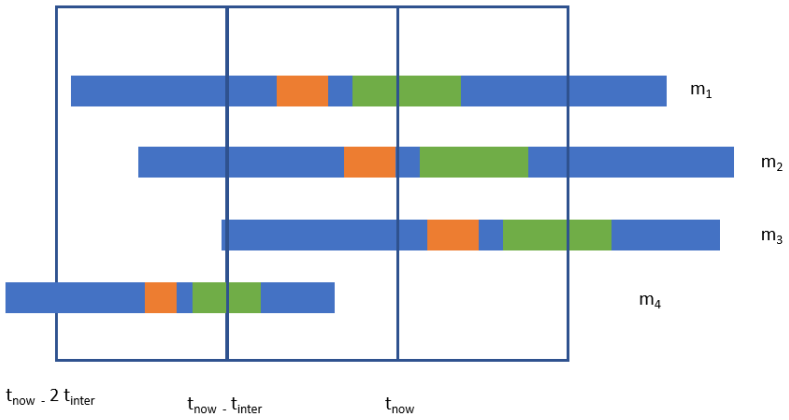


Figure 5.4: The sliding interval over time series data of 4 machines, green indicates healthy data, red indicates unhealthy data and blue indicate unknown, for every sliding interval, the healthy data are used to update the model if no following repair action in the next interval

The online training simulation is summarized as following.

Algorithm 1 Online learning simulation

Input: online training set d_o, model M, buffer interval t_{inter}

Initialize: Perform an offline training on the data before date t and obtain the initial model, $t_{now} = t + 2 * t_{inter}$, the last date in the dataset is t_{last}.

1: **while** $t_{now} \leq t_{last}$ **do**
2: $d_{slide} = \emptyset$
3: **for** machine m in the total set of machines **do**
4: let t_m be the time of the corresponding data points d_m of machine m, t_{m1} are the points that satisfy $t_{m1} \subseteq t_m$ and $t_{now} - 2t_{inter} < t_{m1} - t_{inter} \leq t_{now}$
5: **if** $|t_{m1}| > 0$ and there is a repair action $t_{repair} + 1\ year < t_{m1}$ and no repair action during $t_{now} - t_{inter}$ and t_{now} **then**
6: Add corresponding data d_m of t_{m1} to d_{slide}
7: **end if**
8: **end for**
9: Backpropagation perform on dataset d_{slide}, test the updated model on the data between $t_{now} - t_{inter}$ and t_{now}
10: $t_{now} = t_{now} + t_{inter}$

Notice for each retraining the number of training data can be different.

The online learning baseline model has no retraining process. The difference between the online learning baseline model and offline learning model is the way to divide the dataset for training and testing. Offline model dividing the dataset based on machines while online learning baseline model divided the dataset based on timeline.

5.4 Online detection and offline detection

The difference between online and offline detection can be shown in the following. In offline detection, after obtaining the anomaly scores of all the testing data, a threshold is optimized on the testing data. In online learning, a threshold is optimized on the data other than the testing data.

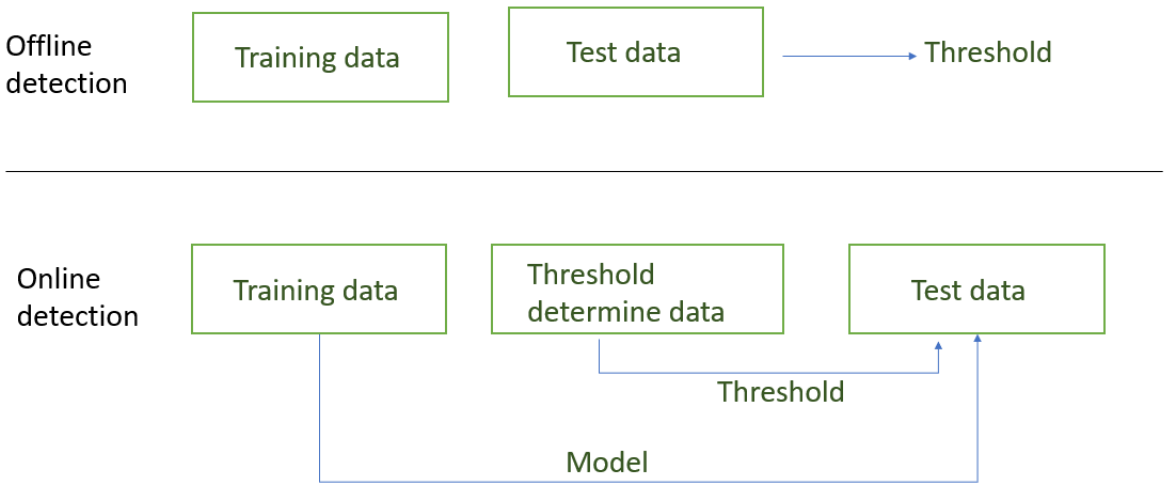


Figure 5.5: Offline and online detection compare

5.5 Overall System Design

The overall machine learning based diagnostics system design can be drawn as the in Figure 5.6. The signal data first go through the detection model. If the output anomaly score is above threshold, then an alarm will raise to inform the monitor system there may be unhealthy behavior occur, and the signal will then go through the second model, which is the classification model. According to the failure mode determined by the model, a corresponding action is taken.

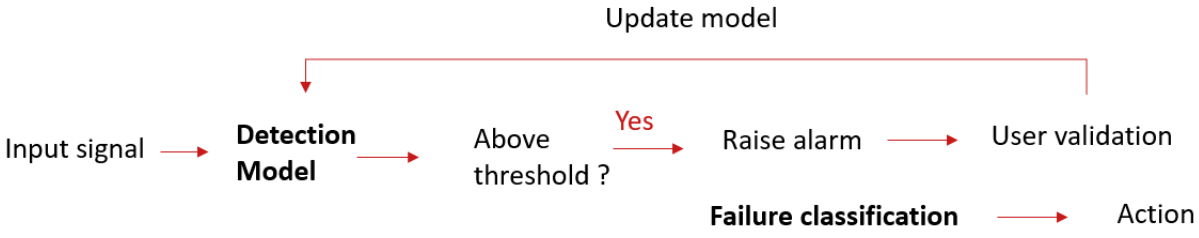


Figure 5.6: Anomaly detection and classification flow chart

The model is expected to take into account the feedback of diagnostic engineers during usage. The feedback would be transformed to structured labeled data for retraining the model. When new repair action occurs, there would be new healthy and unhealthy data and these new data could be used for retraining. Human feedback is needed because in reality the repair action cannot act as the indicator of healthy and unhealthy data perfectly. It is possible that the part is still healthy and functioning well but is swapped for a proactive maintenance. The experienced diagnostic engineers can give a more accurate label on the data. These feedback would be transformed as new training data to update the online learning detection model.

In this study no human labels are available therefore the new data label is derived only from the actions.

Chapter 6

Experiment

There are three methods used for training the anomaly detection Omnianomaly model:

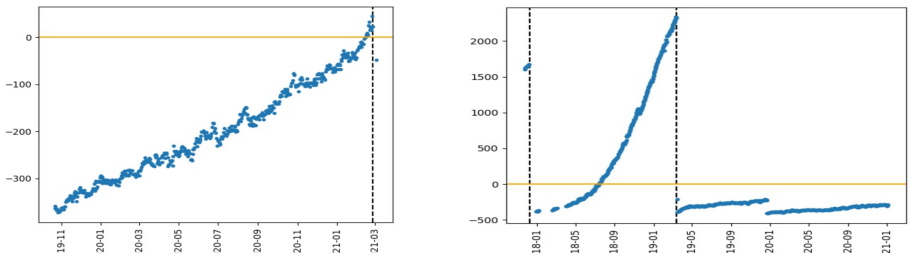
1. Offline training on some machines and offline test on other machines with cross validation
2. Online training on all machines then offline test and online test
3. "Run-to-failure" experiments which train and test on one individual machine.

The offline detection is used to show the capacity of the Omnianomaly model in this prognosis case to see if the model can assign healthy data low anomaly score while giving unhealthy data high anomaly score. Online detection is used to test the model on a more real-life scenario. The "Run-to-failure" experiment is to see if training and testing on one machine provide advantages.

For all the experiments using the Omnianomaly RNN-VAE model, the latent space sample number is set as 1024 to reduce the randomness when sampling from latent space during testing. The latent space dimension is set as 20 unless specified otherwise. Window size of RNN is set as 7. All variables are normalized to a range of 0-1 before feeding into the models.

6.1 Detection model offline training

The testing results of offline learning show that the model can detect degradation trends in some machines, even though the range of the data is different, thus the thresholds to distinguish between healthy and unhealthy can be different in different machines. The degradation trends are also different in different machines, in terms of the slope and change of the slope over time. Sometimes the slopes look linear and sometimes slow at the beginning an increasing faster later. Following are the examples of the anomaly score in two machines.



Machine 1 Machine 2

Figure 6.1: Anomaly score of two sample machines

It is sometimes observed that the anomaly score is going upwards somewhere before the repair action, then it suddenly drops to a lower value before the repair action. An example of this

trend can be found at the following. This could be explained by the machine may be working suboptimally and then a lens set up action or a testing is performed on the machine so the signals drop to a normal value. But the action did not solve the issue so a repair action is still performed later. This indicates that the model result is affected by other actions such as lens set up and the zernike signals cannot be used as the sole indicator for the healthy status of the machine.

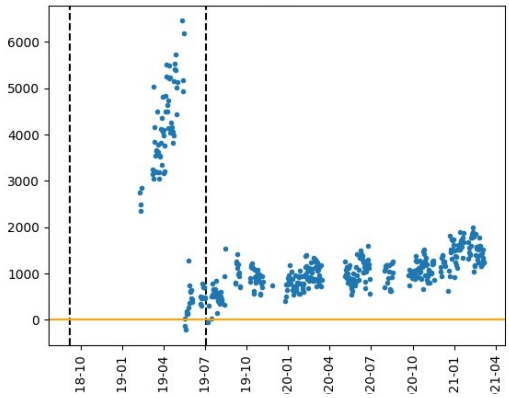
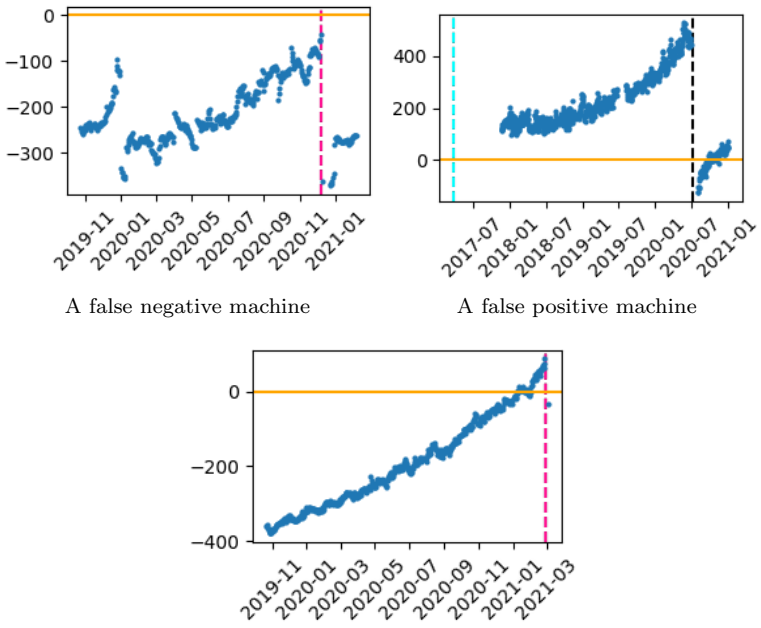


Figure 6.2: A machine with a trend of going up and then drop before repair

When evaluating on the hit rate and quality, each repair action only has one label, either false positive, true positive or false negative. True negative is not directly used for calculating the metrics, thus is not a concern here. A false negative is counted if the selected threshold fails to detect the abnormal data, that is, the abnormal data appears below threshold. False negative action would increase if the threshold is set higher. A false positive is counted if the normal data appear above threshold. A true positive is counted if the abnormal data appears above threshold. When the threshold increases, the true positive rate decreases. A true positive is counted if the data is unhealthy and appear above threshold. When the threshold increases, true positive rate decreases. Example of the false negative, false positive and true positive are shown in Figure 6.3. If it is uncertain whether there is an action during a specific time period or not, then that specific time period is excluded.

Under the above conditions, a threshold can be selected for calculating the hit rate and quality. The hit rate reaches 0.69 when the quality is 0.76 when the threshold is set as 0. The hit rate is 0.20 when the quality is 0.91 when the threshold is 1220. Some of the uncertain actions needed to be further investigated through the machine logbook or consulted the engineers working in the local field. Some human judgement were included when determining the false positive, false negative and true positive in calculating the hit rate and quality. Therefore, these two metrics are not included into every experiment.



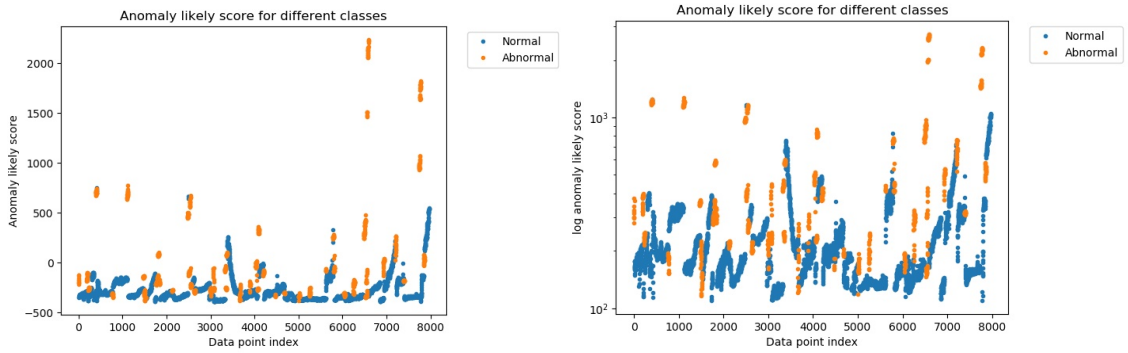
A false negative machine

A false positive machine

A true positive machine

Figure 6.3: Examples of anomaly score of 3 machines when threshold is set as 0, vertical lines indicate repair actions

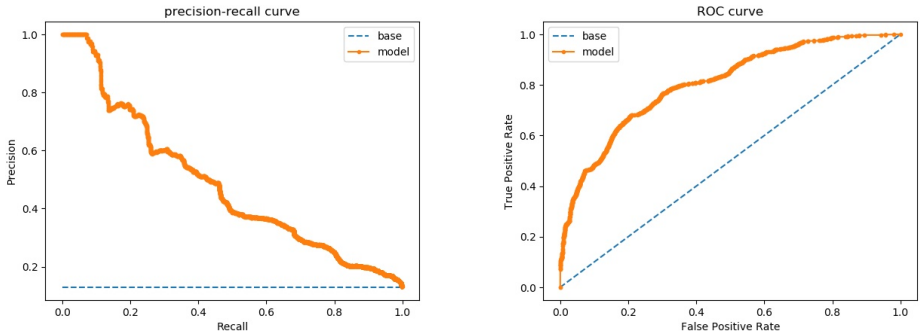
The anomaly scores of all machines concatenated can be shown in the following Figure 6.4. From this visualization, it can be seen that the model can give some anomaly points a higher score compared to the healthy points. Notice in this figure only the labeled points are counted for evaluation. Labeled unhealthy data are 30-7 days before the repair action and labeled healthy data are 15 days - 360 days after the repair action. The reason that the healthy data starts 15 days after is the signals are usually unstable just after the repair action, so the data in this period is excluded.



Anomaly score

Anomaly score with log scale

Figure 6.4: Anomaly score of all points in offline learning setting



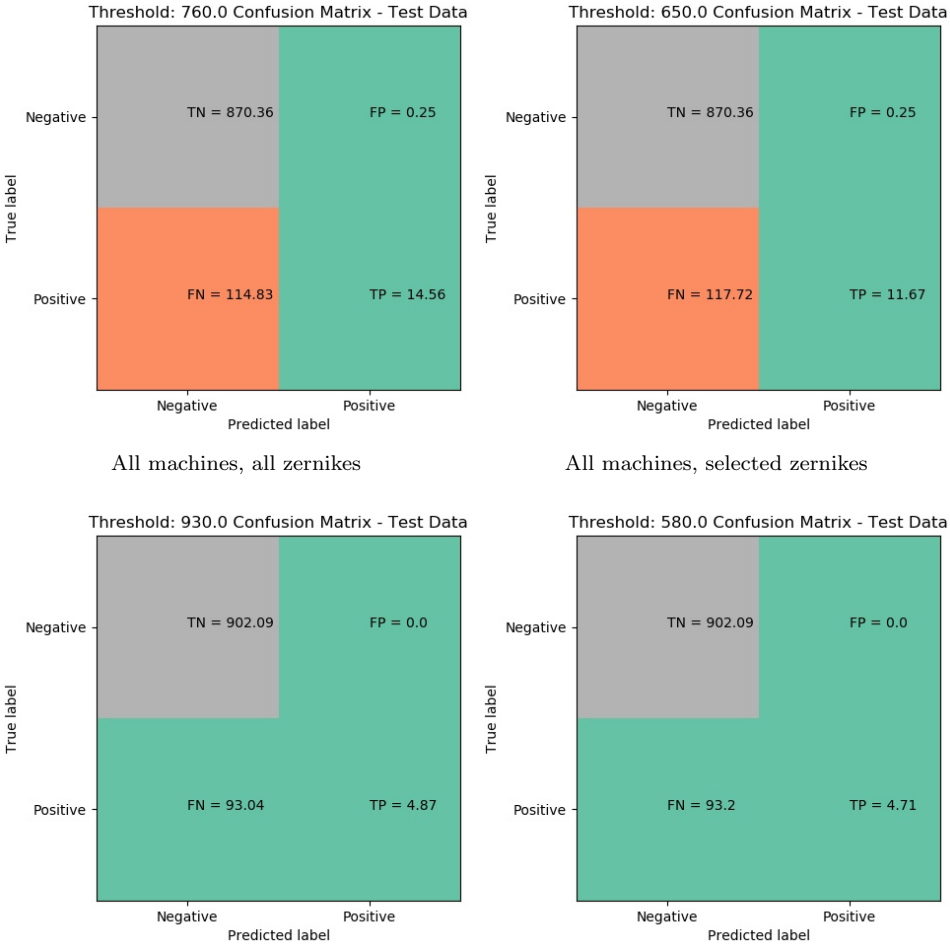
Precision-recall curve Receiver Operating curve

Figure 6.5: Precision-recall curve and Receiver Operating curve of the model, blue dash line indicates a model with no skills

Some zernike coefficients are considered as more significant indicators of WELLE failures by domain experts. To see the effects of incorporating expert knowledge on feature selection, the experiments are performed using all zernike data and the selected zernike data. The selected zernikes are selected by human experts and are considered as the most important features related to WELLE failure. Different types of machines have differences in designs. To find out the whether the type of machine has an impact on the result, the model is trained on all types of machines compared to trained on one specific type of machines. The following are the comparisons of results.

	recall (90% precision) based on data points	recall (90% precision) based on machine	partial AUC ROC	partial precision-recall AUC	Correctly detected machines percent
All zernikes Mixed type	11.3%	6.5%	0.651	0.111	36.4
Expert selected zernikes Mixed type	13.6%	2.3%	0.704	0.127	36.4
All zernikes One type	5.1%	0.0%	0.610	0.051	30.0
Expert selected zernikes One type	29.4%	32.3%	0.676	0.297	22.2

Table 6.1: Offline training results



All machines, all zernikes

All machines, selected zernikes

One type machines, all zernikes

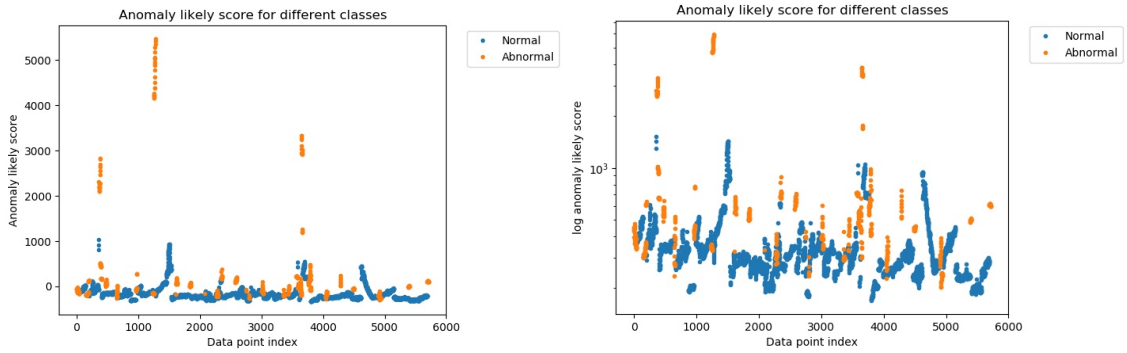
One type machines, selected zernikes

Figure 6.6: Offline training confusion matrix, with precision larger than 97%

Different metrics show different results for 4 different training settings. In terms of partial AUC-ROC and partial precision-recall AUC, training on selected zernike drifts within either one type or mixed all types shows better results than using all zernikes based on data points. Training within one type of machine with all zernikes shows dramatic drop on the performance. When training on one type of machines, the number of machines reduced since the machines are restricted to one specific type. A smaller amount of training data with high dimensions may lead to overfitting more easily, therefore results in worse performance. Reducing the feature dimensions can reduce overfitting. The trade-off between variance and bias exists here. Furthermore, the machine to machine variation in terms of the zernike drifts signals could be larger than the type to type variation. There are other factors such as type of the specific part may affect the result more directly than the commercial type of the machine does. Using selected zernikes on one type of machine shows better results on all metrics expect for correctly detected machine numbers. The result calculated by per data points is generally better than the result calculated by per machine in this experiment. This could be that the correctly detected machines have more data points counted than the machines that are detected wrongly.

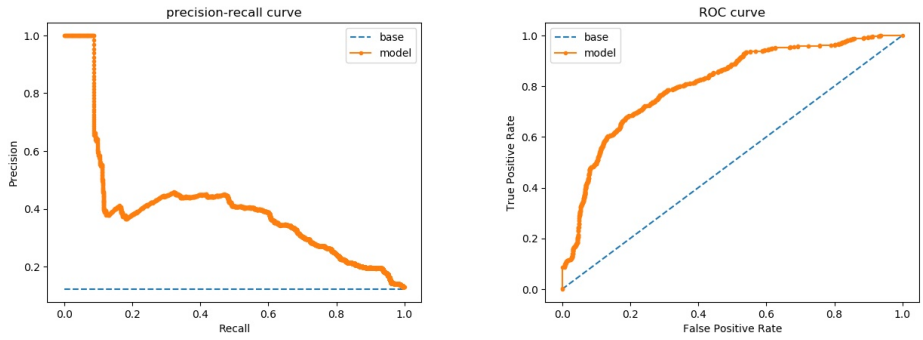
6.2 Detection model with online training

The machine evolves over time and may affect the signal patterns. To allow the model evolve over time and incorporate the feedback of experts, online learning is applied to the model. The same model is then trained using online learning algorithm. The data before 2019 is used for initial training, and the data starts from 2019 is used for incremental training. Three experiments results are compared, shown in the following. The first experiment uses incremental learning only on the new incoming data. The second uses incremental learning on all the available data at each retraining stage. The baseline model is trained only on the data before 2019, and no incremental learning is applied to this model. All zernike variables are used for online training.



Anomaly score | Anomaly score with log scale

Figure 6.7: Anomaly score of all points in online learning setting



Precision-recall curve | Receiver Operating curve

Figure 6.8: Precision-recall curve and Receiver Operating curve of the model using incremental learning, blue dash line indicates a model with no skills

	recall (90% precision) based on data points	recall (90% precision) based on machine	partial AUC ROC	partial precision-recall AUC	Correctly detected machines percent
Online learning with new data	19.3%	18.9%	0.671	0.211	41.9
Online learning with all available data	24.8%	2.9%	0.652	0.239	41.9
Online learning baseline model	36.2%	27.0%	0.705	0.356	51.2

Table 6.2: Online and offline training results, using one threshold for all testing data

From the evaluation metrics, the two incremental learning methods do not show improvement when compared to a baseline model. One of the reason that the incremental learning on new incoming data does not show improvement could be that in each training, the amount of training data is too small. Another reason could be that the scale of the data changed after each retraining, but at the end only one threshold is selected for all data, so the fixed threshold cannot fit with the changing scale of the data. In the setting of online learning with only new incoming data, the quality of the new training data may be different from the past data and overall the result is unstable when observing the testing result in machine individually. Training with all available data does not show much improvement compared to training only on the new incoming data. Figure 6.9 is the testing result of the online learning baseline model with the optimized threshold.

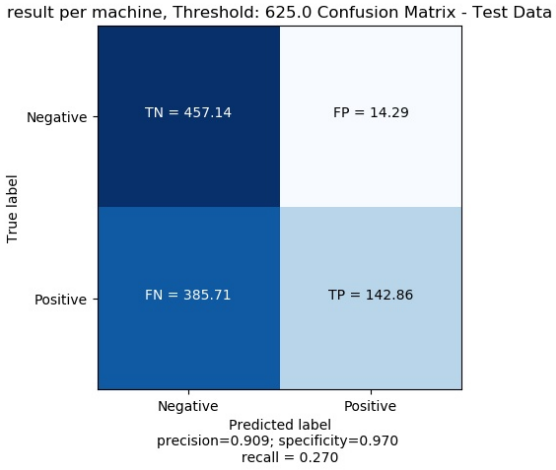
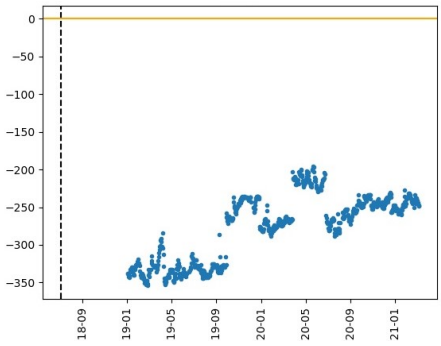
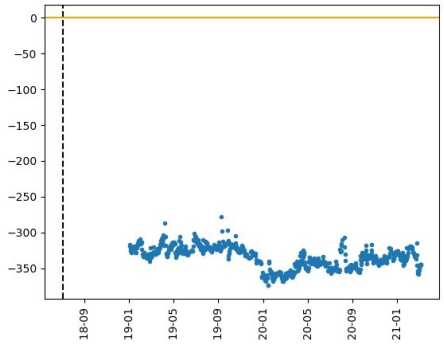


Figure 6.9: Online learning baseline mode offline detection result

In some cases, training using all available data provides smoother curves as shown in Figure 6.10
.



A machine with incremental learning on new incoming data machines



Same machine with incremental learning on all available data

Figure 6.10: Online learning on only new incoming data and online learning with all available data

6.3 Detection model trained on one machine

The "run-to-failure" experiments which trained and tested on the same machine are performed on a few selected machines. These machines satisfy the criteria that they have repair an action at a later period but not the earlier period. The first half of the data is used as training data and the later half is used as testing data.

The result anomaly score shows growing trend before repair, but does not drop and even increase after repair action. This means the pattern of training data is different from the healthy data after repair. The training data lies in the period of 0.5-2 years before repair action and the degradation is likely to have occurred during this period, thus they cannot represent the real healthy data.

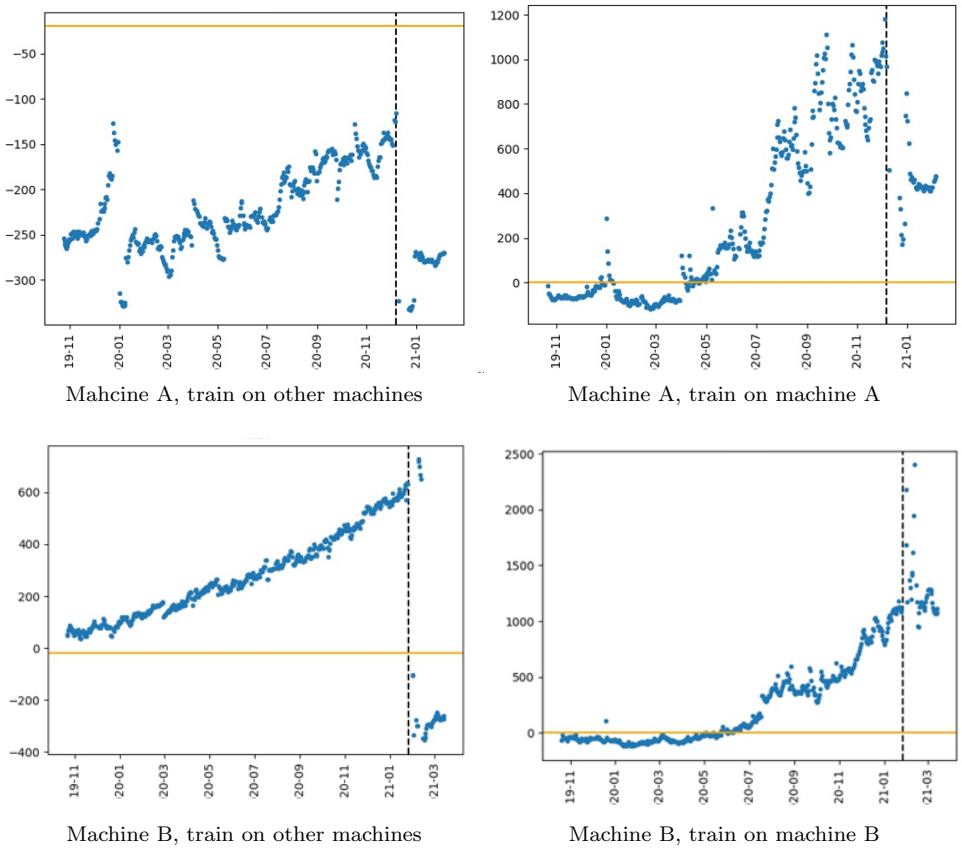
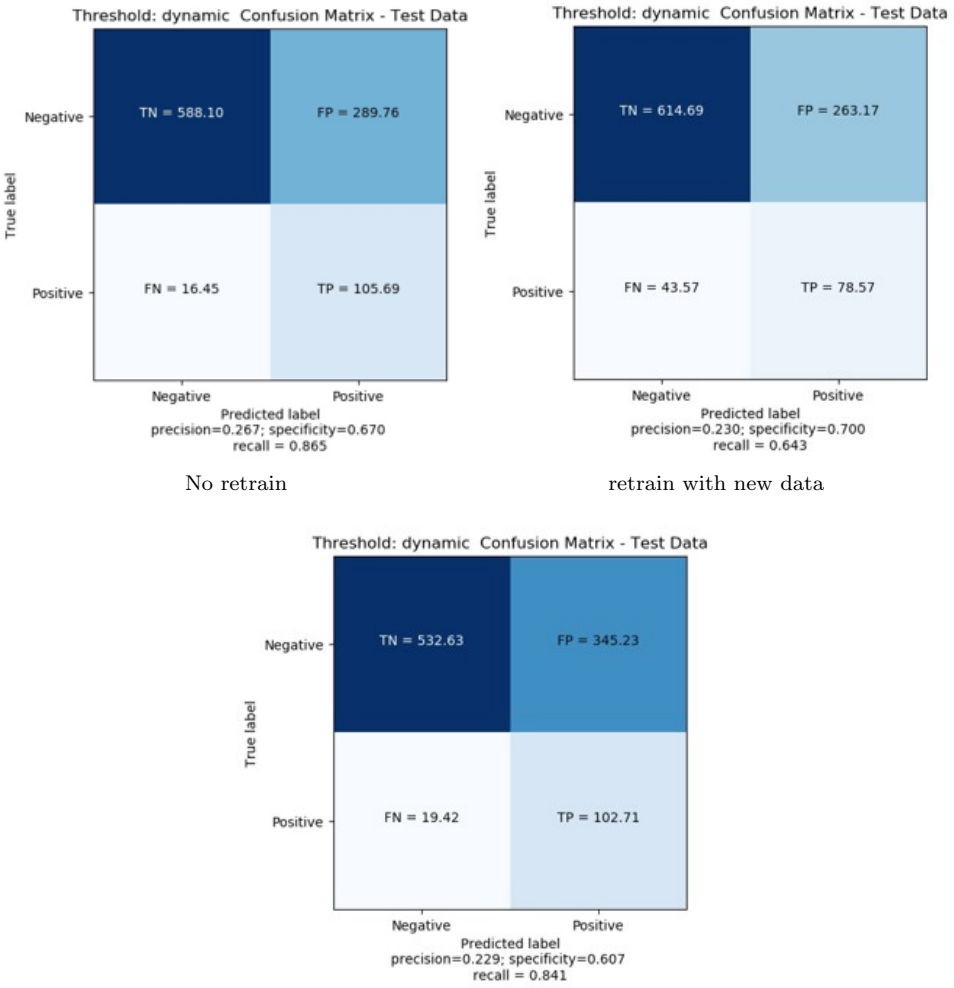


Figure 6.11: Training on other machines and training on machine itself comparison

Comparing training on healthy data of other machines and training on machine itself, there are 3 observations: 1) In both training methods, the model is able to detect the upward trend of anomaly score before the repair actions 2) After the repair action, the anomaly score from the model trained on the former method drop to a lower point while the anomaly score trained on latter method do not appear to drop to lower than beginning 3) When training on other machines, the threshold that distinguishes between anomaly and healthy data need to be adjusted according to the test machine. In Figure 6.11 (a) and (b) are two testing results of two machines, both from models trained on healthy data of other machines. Machine A needs a threshold lower than 0 while machine B needs a threshold higher than 0.

6.4 Detection model online learning with online detection

All the above testing are offline detection, which means during the testing phase, the anomaly scores of the test data are calculated, and then a threshold is optimized for these scores. When the model is used for detecting anomalies in production, a threshold needs to be preset to determine the current healthy status of the machine. To determine the threshold, a separate set of data is used. So each time when training the model, there are 3 set of exclusive data, including training data, testing data and a dataset for determining the threshold. In the following experiment, the data before 2018 are used as threshold-determine data. Each time after retrain the model, a new threshold is optimized using this set of data, so the threshold is dynamically changing.



No retrain

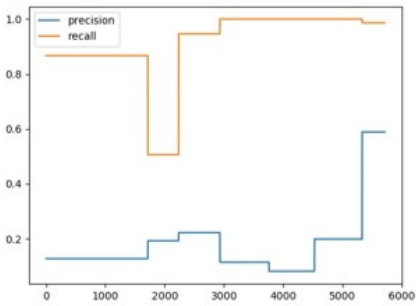
retrain with new data

Retrain with new and old data

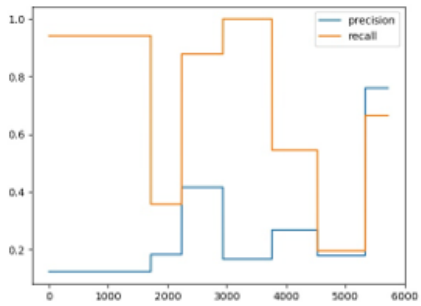
Figure 6.12: Confusion matrices of a baseline model and two online learning methods with online detection

It can be seen that the precision is quite low and could not meet the requirement of this business case. The threshold derived from the separate set is apparently too low. One of the reason that the threshold is too low could be that the dataset to determine the threshold is close to the training data and the pattern of it would be similar to training data. Therefore, the derived threshold would be low, as the model is trained to minimize the loss on the training dataset.

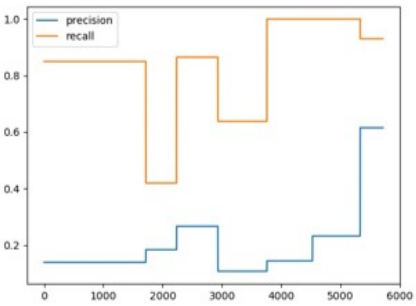
Even though there is no retraining in the baseline model and the threshold keeps constant, the performance of the model keeps changing. The reason could be that over time the dataset is changing, so the pattern of the data is changing as well as the number of healthy and unhealthy data. The unstable performance of the model is another reason.



No retrain



retrain with new data



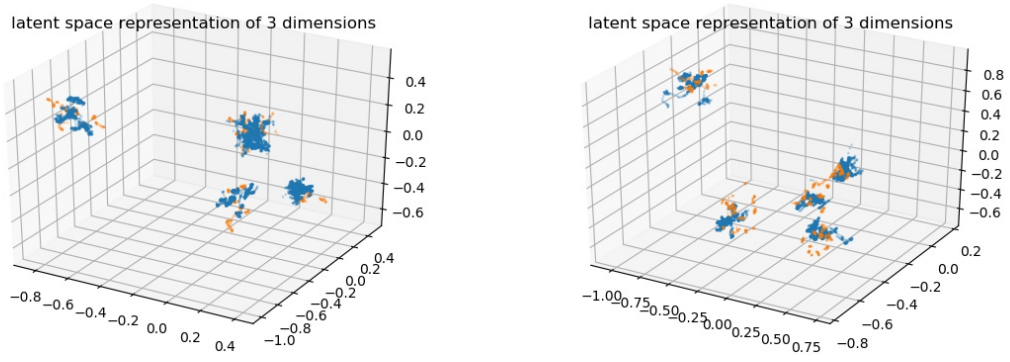
retrain with new and old data

Figure 6.13: Precision and recall over time for a baseline model and two online learning methods with online detection

For both training with new data and training with old and new data, the precision and recall do not show consistently improvement over time. This could be that the retraining is making the model harder to converge. And when training with only new data, the catastrophic forgetting can occur.

6.5 Latent space visualization

To see the latent variable behaviors in different runs of experiments, the latent space are visualized. The 234 features are mapped to a 20 dimension latent space by the detection model. Each feature has a mean and variance. The mean of each feature is visualized. To visualize the behaviors of the latent space, t-distributed stochastic neighbor embedding (t-SNE) [73] is applied to visualize all 20 dimensions. Some latent variable features are sampled for visualization. From a few latent variable representations, it seems that data are clustered into different groups in latent space. Notice cross validation is used for training the offline detection model, and it is a 5-fold cross validation. So there are 5 trained models in total. It is likely each time the data has learned a different mapping function.



3 latent variables Another 3 latent variables

Figure 6.14: Latent space representation (Blue: Normal data, Red: Abnormal data)

From the t-SNE plot below there is no apparent distinguish between normal and abnormal data. Overall the data seems scatter and no apparent cluster presents.

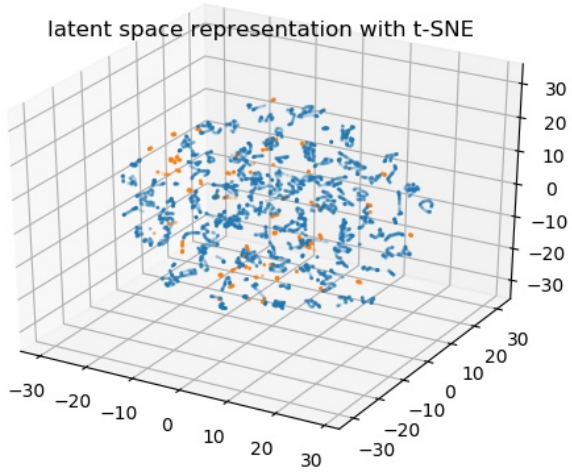


Figure 6.15: Anomaly score of healthy data and unhealthy data of all machines (Green: Normal data, Red: Abnormal data)

To allow visualizing all latent dimensions in the latent space in a 3D-plot, the following experiments are conducted under the restriction that the latent space of the variational autoencoder is set to 3 and the dataset is the expert zernike dataset which has 64 features. The model still keep its performance under these settings, in terms of the evaluation metrics.

The 5-fold cross validation results in 5 variational autoencoder models. A visualization of all the latent space variables of all machines is shown below. It is obvious that each time training the model on a different dataset the latent space moved.

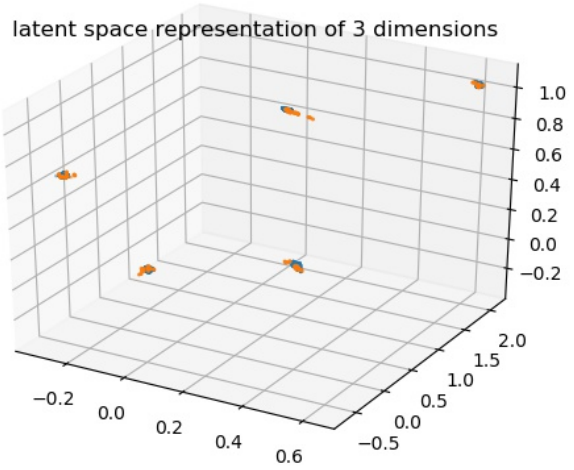
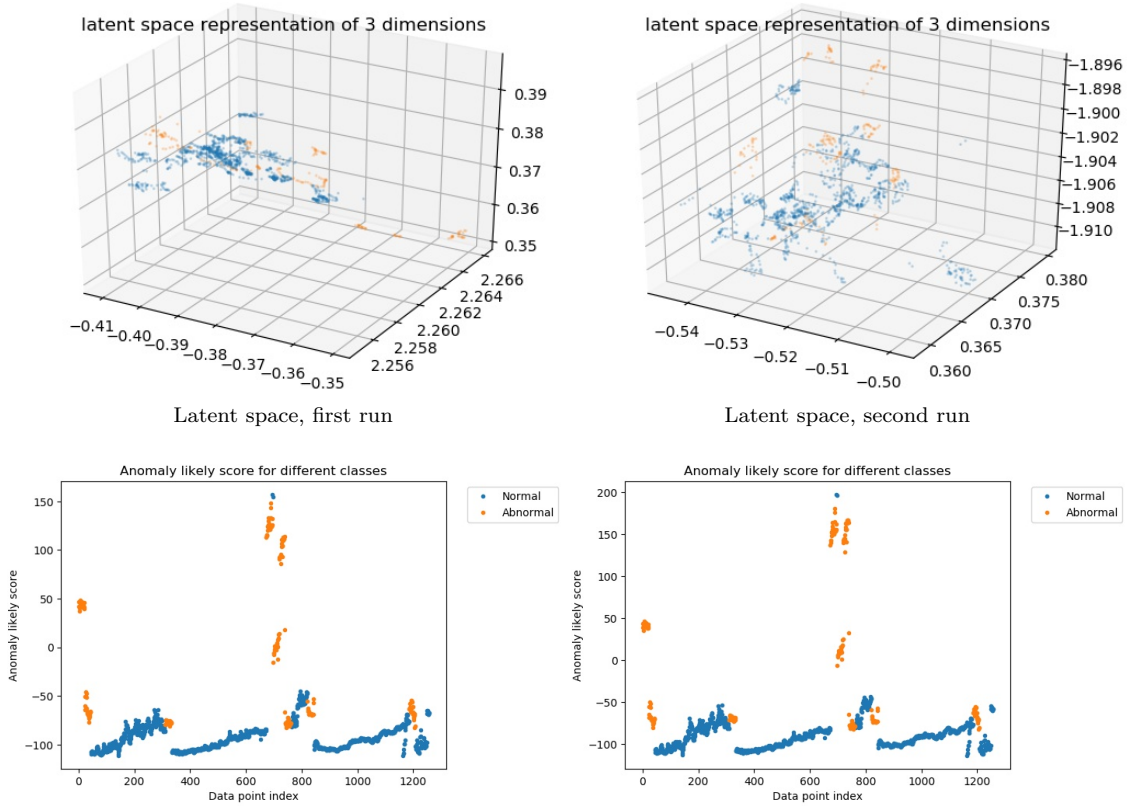


Figure 6.16: Latent space visualization of testing data

Neural networks are a black box model, so it is not clear how they map the data to latent space and then reconstruct the data, but it is reasonable to assume that there is no order in between the latent space variables. As all the layers are dense layer, suppose the neural network learns three features and have three nodes at the bottleneck, the three features can appear in any order in these three nodes, even if it learns the exactly 3 same features each time.

The following experiment selects one fold of data from the 5-fold cross validation dataset to exclude the difference between different dataset and see the effect of different runs from same dataset. When training the model on exactly the same dataset, different runs result in different area in latent space and different range of anomaly score, even though the result reconstruction likelihood graphs on testing data look similar. This different latent space area is reasonable as even though each time the same 3 features are extracted from all dimensions, there is no order between these extracted features. There are many stochastic steps inside the model, this could explain different runs result in different score scales.



Latent space, first run

Latent space, second run

Reconstruction likelihood result, first run

Reconstruction likelihood result, second run

Figure 6.17: Latent space and reconstruction likelihood on testing data of 2 runs

Mapping the data to different latent space after each run could bring challenge on retraining the model on the new data, as the model may map the data to a different latent space area after each retrain. The memory of the old training data may lose, or the relationship between the old training data and new training data may be missing.

The changing of the score scale also brings challenge, as a new threshold needs to be determined after each retrain. The changing of the scale between different runs is much less compared to the change of the scale between different machines in the same run.

6.6 Classification model

The classification models are a set of basic machine learning models. These models are benchmarked individually. Two additional experiments are using tree-based model to select the important features and then use neural nwtworks for classification based on these features.

The failure classification uses "leave-one-out" cross validation for evaluation, so the total number of experiment runs are 42 as there are 42 machines in this dataset in total. The proportion of repairing failure type A issue is 0.53. A random guessing classifier should reach at least 50% accuracy. There are two subsets of zernikes considered as more related to the failure type A and failure type B issue, respectively. The first subset contains 64 zernike drifts and the second subset contains 46 zernike drifts. There are overlap in these 2 subsets and the union of these 2 subsets contains total 94 zernike drifts.

The degradation trend usually starts at least a few months before the repair action. The unhealthy data is defined as 1-60 days before repair action. Notice even each machine has more

than one data point as unhealthy data, usually these unhealthy data in the same period of same machines are statistically dependent and similar so may be viewed as same data.

The max depth of the decision tree is optimized through brute force and is set as 5. The neural networks are optimized using grid search method. The final neural networks applying on the extracted features by decision tree or XGB contain 2-4 hidden layers, each with 10-30 nodes. Dropout and batch normalization are applied in between any two layers to reduce overfitting.

Accuracy	Decision tree	XGBoost	lightGBM	Fully connected NN	logistic regression	SVM	Decision tree + NN	XGB + NN
234 zernikes	61%	44%	46%	61%	61%	62%	68%	**75%**
expert zernikes	51%	63%	62%	62%	60%	62%	**75**%	66%

Table 6.3: Failure classification model

The results show that in this small dataset, when feature dimension is large, decision tree performs better than its ensemble counterparts and when feature size is reduced to below 100, ensemble methods perform better. A shallow fully connected neural network did not show advantages when all 234 zernike features are used, possibly because it is easier to become overfit than the decision tree model, especially with this small amount of data.

Among the 42 decision trees built by the 42 runs of the experiments using all zernike variables, there are 15 features having cumulative feature importance more than 0.5. 9 out of these 15 features are also selected by experts as the expert zernikes.

The loss of neural networks is cross entropy loss. During the training of neural networks, the validation loss starts to increase at the early epochs and the validation accuracy does not decrease. Sometimes the validation loss increases as validation accuracy increases. This could be that the model is making more correct predictions while the confidence of these predictions decrease.

The accuracy of neural networks trained with features selected by tree-based models is affected by the number of features used, so this would be an additional parameter to fine-tuned when utilizing the tree-based selected features for neural networks. The following Table 6.4 shows the relationship between number of top features used and the corresponding accuracy. The table uses the features selected by Extreme gradient boosting machine among the 234 zernikes features. The zernike features selected by tree-based are shown in section A.1.

Number of features	9	11	13	15	17	19	21	23	25
Accuracy	67%	68%	69%	70%	**75%**	68%	69%	72%	65%

Table 6.4: Failure classification model performance when using different number of selected features

The confusion matrix of the best classification model is shown at the following.

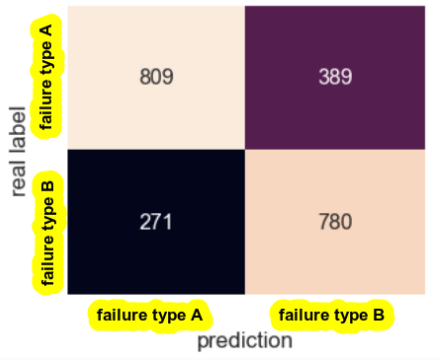


Figure 6.18: Confusion matrix of the classification model using XGB + neural networks

The classification model would be used after the failure is detected in the first model. Therefore, its performance is affected by the first model. If the detection is inaccurate, then it would be meaningless to use the classification based on the detected anomaly. So it is more important to ensure the performance of the first model, therefore more focus is put on the detection model. However, classification is by itself important, as when failure occurs, classifying the failure mode is an unavoidable task.

Chapter 7

Conclusions and future work

The Omnianomaly model is able to detect the degradation trends of machines. The model trained on part of the machines is able to detect the monotonic upward trends on some other machines while failed to detect the trends on some other machines. The anomaly score range of different machines varied. This means different machines need different thresholds to distinguish between healthy and unhealthy data. This brings challenges for the failure prediction, as enough data needs to be collected in a machine for determining the specific threshold for that machine.

Using all zernike features for the detection model yields better result on a machine base while using a selected subset of zernike gives better result on a data point base. Using the model on one type of machine instead of mixed machine type gives worst performance on the result when all zernike variables are used. The reason may be that the machine to machine variance is larger than the type to type variance and the machine type is not the most important factor to cluster machines. When all zernike variables are used, training each model for each commercial type does not help improving the performance. But when only a subset of selected zernikes are used, training the model on one type of machine seems to improve the results of most evaluation metrics.

For both offline detection and online detection, incremental learning using only the new incoming data does not show improvement compared to the baseline model that does not use incremental learning. Even using all available data for incremental learning does not seem to be improving in terms of the evaluation metrics. When online detection is applied, the precision is too low to satisfy the business requirement. Apparently the threshold is too low for the testing data and this leads to low precision. Overall, retraining the model with new data in this case does not show promising result.

The online learning baseline model performs better than the offline learning model, which trained on some machines and tested on other machines. This means if training data contain the historical data of the testing machines, the results are better. The detection model is more suitable to be utilized for machines that already have historical data and may be less suitable for new machines with no history data recorded.

The VAE model shows capacity for offline detection but when it comes to online detection, the challenge of selecting a threshold beforehand pose difficulties for using the model.

When using a high dimension with a small data amount dataset used in this research, the decision tree is able to select some important features for improving the result of the neural network model. The combination of extracting features using tree-based method and utilizing these features in training with neural networks yields better result than using only decision tree or neural networks alone on the 234 zernike variables. The number of features used for neural networks needs to be fine-tuned to obtain a satisfied result. More than half of the features selected by tree-based method are actually selected as the significant failure indicators of the 2 type of failures. The above results could indicate that the tree-based method can be a useful feature selection tool in this failure classification case. The number of zernike features used have a significant impact on the neural network classifier. Only 17 out of 234 zernike variables are used in this case.

7.1 Business Value

The detection model with online detection does not show its business value due to the low precision. The following business value assumes the online detection can reach the performance of offline detection, so it can reduce the overall down time of the machines. The business value of the detection model is calculated under the previous assumption indicated in the introduction chapter. The detection model can predict 27% cases. The false positive is 1.429% of the total output and true positive is 14.29% of the total output. The downtime saved is:

$$\begin{aligned}&Total\ downtime\ saving\\&= 14hr * 0.5/machine/year * 1000\ machine * 27\% -\\&1.429\%/14.29\% * 0.5/machine/year * 24 * 1000\ machine * 27\%\\&= 1566hr/year\end{aligned}$$

The business value of the classification model can be calculated as the following:
Total labor hour saved:

$$0.25\ labor\ hr * 1000\ machines * 0.5/year = 125\ labor\ hr/year$$

7.2 Comparison with previous work

Following is the comparison with the work conducted by previous students.

One previous student applied autoencoder with LSTM for anomaly detection [74]. The testing data used is 7-1 days before the repair actions. In this research focuses on prediction so the testing data is 30-7 days before the repair action. The machine dataset is also quite different although there might have some overlap. In the previous research, through optimizing a threshold on the testing data, the sensitivity reached 36% when the precision is 1. In this research the offine training result is 13.6% sensitivity (recall) with a higher than 90% precision. The online training baseline and online testing model gives 36.2% recall with a higher than 90% precision. The result is based on data points not based on actions.

Content	Previous work	This work
Model	LSTM-Autoencoder	OmniAnomaly
Anomaly data used	7-1 days before repair	30-7 days before repair
Train and test on different machines	Precision: 100% Recall: 36%	Precision: >90% Recall: 13.6%
Online learning baseline	-	Precision: >90% Recall: 36.2%

Table 7.1: Comparison with previous work on offline detection

In this study, an additional metric based on per action is introduced and evaluation of each machine individually is done in some experiments. Inspecting on each machine individually can give more insights on the result and the degradation trends can also be observed.

Online detection is introduced in this study to examine the model performance for anomaly prediction for a more realistic evaluation in terms of usage of the model.

Online learning is introduced in this study to investigate whether the model can take feedback from users and update over time to improve the performance.

A failure classification model is built for anomaly classification and shows promising result.

Furthermore, a dashboard is developed for dynamically visualizing the results and incorporating human decisions in evaluation, such as adjusting the threshold and input evaluation for the testing result, as shown in section C.1.

7.3 Future Work

Both the detection model and the failure classification would need to be further improved before putting into production.

The detection model could be further improved by tackling the irregular time intervals with special RNNs such as NeuralODEs[75] and ODE-RNN[76], or by simply encoding the timestamp into the dataset.

The detection model is a one-class classifier. The benefit of this type of model is that it only needs healthy data for training. But in the dataset we also have some unhealthy data available, so adding these unhealthy data into training could be a way to improve the model. A one-class classifier would view any outlier as unhealthy data, but this may not always the case. Adding the unhealthy data to the VAE model and train the model to minimize the reconstruction likelihood of the unhealthy may help further distinguish the healthy and unhealthy data. The unsupervised detection model can also be turned into a semi-supervised model by incorporating a classifier, although the unbalanced dataset could pose a challenge for the classifier.

Adding some sub-healthy data as healthy data into training could be an idea to make the model less sensitive and could possibly reduce the false positive rate.

It could be beneficial to also predict the interval before a downtime occurs or the remaining useful time. This could be an additional model using the anomaly score as input, or could use the original signal data as input. It is also possible to fit a polynomial or exponential function to predict the remaining time. But based on the trends shown in the experiments the trends are not homogeneous so fitting in one function could be difficult.

Adding additional rules such as unhealthy alarm would raise only if there are a certain amounts of points in a row higher than the threshold, or using the moving average instead of making a decision based on each point may further reduce the false alarm rate.

More variables that have effects on the result could be taken into consideration for training the model, such as the type of the part, LoCo data and initial setup value.

Other machine learning models such as Baysian network or clustered models that can output the probability between 0-1 can be considered to avoid the problem of uncertain output range.

More domain knowledge could be incorporated with the model to improve the model, especially for the explainability of the model. But this would usually mean there needs to be persons who have both domain knowledge and understanding of machine learning models.

As the VAE model poses the challenge that the scale of the anomaly score changed randomly, other machine learning models may be considered, such as Bayesian network and cluster models. Those models can provide output as probability between 0 and 1 and could avoid the uncertain scale problem.

7.4 Recommendation

It is recommended to record each repair or adjustment action in a more standard format in a centralized database. Therefore, more accurate data can be retrieved for training and evaluating machine learning models more reliably. The recorded data need to be more consistent and complete.

Other relevant data could be included in the model, as the zernike drift signals still have some limitations on predicting the healthy status of the machine. If only use the zernike signals as training, other actions that could affect zernike signals should be included in the dataset.

More expert knowledge can be incorporated into the model developing process for improving and validating the model.

Some zernike drifts contain more important information towards the result and these zernike variables can be focused on when providing the input for the machine learning models.

Bibliography

[1] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2828–2837, 2019. i, v, 12, 15

[2] Mark A Van de Kerkhof, Wim de Boeij, Haico Kok, Marianna Silova, Jan Baselmans, and Marcel Hemerik. Full optical column characterization of duv lithographic projection tools. In *Optical Microlithography XVII*, volume 5377, pages 1960–1970. International Society for Optics and Photonics, 2004. 6

[3] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly Detection: A Survey. *ACM Comput. Surv.*, 41(3), July 2009. 7

[4] H. Izakian and W. Pedrycz. Anomaly detection in time series data using a fuzzy c-means clustering. In *2013 Joint IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS)*, pages 1513–1518, 2013. 7, 9

[5] Longyuan Li, Junchi Yan, Haiyang Wang, and Yaohui Jin. Anomaly detection of time series with smoothness-inducing sequential variational auto-encoder, 2021. 7

[6] Mohamed Elforjani and Suliman Shanbr. Prognosis of bearing acoustic emission signals using supervised machine learning. *IEEE Transactions on industrial electronics*, 65(7):5864–5871, 2017. 8

[7] Jagath Sri Lal Senanayaka, Huynh Van Khang, and Kjell G Robbersmyr. Autoencoders and recurrent neural networks based algorithm for prognosis of bearing life. In *2018 21st International Conference on Electrical Machines and Systems (ICEMS)*, pages 537–542. IEEE, 2018. 8, 11

[8] G. U. Yule, David F. Hendry, and Mary S. Morgan. *Why do we sometimes get Nonsense Correlations between Time-Series? (Journal of the Royal Statistical Society, vol. 89, 1926, pp. 2–9, 30–41)*, page 141–158. Cambridge University Press, 1995. 8

[9] Eugen Slutzky. The summation of random causes as the source of cyclic processes. *Econometrica*, 5(2):105–146, 1937. 8

[10] James H Stock and Mark W Watson. Vector autoregressions. *Journal of Economic perspectives*, 15(4):101–115, 2001. 8

[11] Rui Li, Wim J.C. Verhagen, and Richard Curran. A comparative study of data-driven prognostic approaches: Stochastic and statistical models. In *2018 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 1–8, 2018. 9

[12] Peihua Qiu. *Introduction to statistical process control*. CRC press, 2013. 9

[13] Walter Andrew Shewhart. *Economic control of quality of manufactured product*. Macmillan And Co Ltd, London, 1931. 9

[14] E. S. Page. Continuous inspection schemes. *Biometrika*, 41:100–115, 1954. 9

[15] S. W. Roberts. Control chart tests based on geometric moving averages. *Technometrics*, 1(3):239–250, 1959. 9

[16] Mia Hubert, Michiel Debruyne, and Peter J. Rousseeuw. Minimum covariance determinant and extensions. *WIREs Computational Statistics*, 10(3), Dec 2017. 9

[17] Dongil Kim, Pilsung Kang, Sungzoon Cho, Hyoung-joo Lee, and Seungyong Doh. Machine learning-based novelty detection for faulty wafer detection in semiconductor manufacturing. *Expert Systems with Applications*, 39(4):4075–4083, 2012. 9

[18] Monowar H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita. An effective unsupervised network anomaly detection method. In *Proceedings of the International Conference on Advances in Computing, Communications and Informatics*, ICACCI '12, page 533–539, New York, NY, USA, 2012. Association for Computing Machinery. 9

[19] Warusia Yassin, Nur Izura Udzir, Zaiton Muda, Md Nasir Sulaiman, et al. Anomaly-based intrusion detection through k-means clustering and naives bayes classification. In *Proc. 4th Int. Conf. Comput. Informatics, ICOCI*, volume 49, pages 298–303, 2013. 9

[20] Tran Manh Thang and Juntae Kim. The anomaly detection by using dbscan clustering with multiple parameters. In *2011 International Conference on Information Science and Applications*, pages 1–5. IEEE, 2011. 9

[21] Vijay Kotu and Bala Deshpande. Chapter 13 - anomaly detection. In Vijay Kotu and Bala Deshpande, editors, *Data Science (Second Edition)*, pages 447–465. Morgan Kaufmann, second edition edition, 2019. 9

[22] J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973. 9

[23] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, page 93–104, New York, NY, USA, 2000. Association for Computing Machinery. 9

[24] M. Vafadar and H. Ghassemian. Hyperspectral anomaly detection using modified principal component analysis reconstruction error. In *2017 Iranian Conference on Electrical Engineering (ICEE)*, pages 1741–1746, 2017. 9

[25] J. A. Jablonski, T. J. Bihl, and K. W. Bauer. Principal component reconstruction error for hyperspectral anomaly detection. *IEEE Geoscience and Remote Sensing Letters*, 12(8):1725–1729, 2015. 9

[26] Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, July 2001. 9

[27] Yingchao Xiao, Huangang Wang, Wenli Xu, and Junwu Zhou. Robust one-class svm for fault detection. *Chemometrics and Intelligent Laboratory Systems*, 151:15–25, 2016. 9

[28] Mennatallah Amer, Markus Goldstein, and Slim Abdennadher. Enhancing one-class support vector machines for unsupervised anomaly detection. In *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, ODD '13, page 8–15, New York, NY, USA, 2013. Association for Computing Machinery. 9

[29] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 37–49. JMLR Workshop and Conference Proceedings, 2012. 10

[30] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1705–1714, 2019. 10, 15

[31] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, MLSDA'14, page 4–11, New York, NY, USA, 2014. Association for Computing Machinery. 10

[32] Jagath Sri Lal Senanayaka, H. Van Khang, and K. Robbersmyr. Autoencoders and recurrent neural networks based algorithm for prognosis of bearing life. *2018 21st International Conference on Electrical Machines and Systems (ICEMS)*, pages 537–542, 2018. 10

[33] Yuta Kawachi, Yuma Koizumi, and Noboru Harada. Complementary set variational autoencoder for supervised anomaly detection. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2366–2370, 2018. 10

[34] Sarah M. Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, and Christopher Leckie. High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, 58:121–134, 2016. 10

[35] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985. 10

[36] Yong Shi, Minglong Lei, Rongrong Ma, and Lingfeng Niu. Learning robust auto-encoders with regularizer for linearity and sparsity. *IEEE Access*, 7:17195–17206, 2019. 10

[37] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Icml*, 2011. 10

[38] Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011. 11

[39] Eugen Slutzky. The summation of random causes as the source of cyclic processes. *Econometrica*, 5(2):105–146, 1937. 11

[40] Alan Julian Izenman. *Linear Discriminant Analysis*, pages 237–280. Springer New York, New York, NY, 2008. 11

[41] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. *Logistic regression*. Springer, 2002. 11

[42] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986. 11

[43] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991. 11

[44] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001. 11

[45] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000. 11

[46] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 11

[47] Xiaohang Jin, Yi Sun, Zijun Que, Yu Wang, and Tommy W. S. Chow. Anomaly detection and fault prognosis for bearings. *IEEE Transactions on Instrumentation and Measurement*, 65(9):2046–2054, 2016. 11

[48] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, KDD '18, page 387–395, New York, NY, USA, 2018. Association for Computing Machinery. 12

[49] B. Zong, Q. Song, Martin Renqiang Min, Wei Cheng, C. Lumezanu, Dae ki Cho, and H. Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *ICLR*, 2018. 12

[50] Daehyung Park, Yuuna Hoshi, and Charles C. Kemp. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder, 2017. 12

[51] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014. 12

[52] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 12

[53] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985. 12

[54] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 12

[55] Weixin Luo, Wen Liu, and Shenghua Gao. A revisit of sparse coding based anomaly detection in stacked rnn framework. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 341–349, 2017. 12

[56] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998. 12

[57] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. 12

[58] Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019. 13

[59] Jinwon An and S. Cho. Variational autoencoder based anomaly detection using reconstruction probability. 2015. 13

[60] Karol Gregor, George Papamakarios, Frederic Besse, Lars Buesing, and Theophane Weber. Temporal difference variational auto-encoder. *arXiv preprint arXiv:1806.03107*, 2018. 14

[61] Genshiro Kitagawa and Will Gersch. *Linear Gaussian State Space Modeling*, pages 55–65. Springer New York, New York, NY, 1996. 14

[62] Nicolas Chopin and Omiros Papaspiliopoulos. *Linear-Gaussian State-Space Models*, pages 73–80. Springer International Publishing, Cham, 2020. 14, 15

[63] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2016. 15

[64] Óscar Fontenla-Romero, Bertha Guijarro-Berdiñas, David Martinez-Rego, Beatriz Pérez-Sánchez, and Diego Peteiro-Barral. Online machine learning. In *Efficiency and Scalability Methods for Computational Intellect*, pages 27–54. IGI Global, 2013. 16

[65] Steven CH Hoi, Jialei Wang, and Peilin Zhao. Libol: A library for online learning algorithms. *Journal of Machine Learning Research*, 15(1):495, 2014. 16

[66] Doyen Sahoo, Quang Pham, Jing Lu, and Steven C. H. Hoi. Online deep learning: Learning deep neural networks on the fly, 2017. 16

[67] Claudia Beleites, Ute Neugebauer, Thomas Bocklitz, Christoph Krafft, and Jürgen Popp. Sample size planning for classification models. *Analytica Chimica Acta*, 760:25–33, Jan 2013. 16

[68] Kitsuchart Pasupa and Wisuwat Sunhem. A comparison between shallow and deep architecture classifiers on small dataset. In *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pages 1–6, 2016. 16

[69] Shuo Feng, Huiyu Zhou, and Hongbiao Dong. Using deep neural network with small dataset to predict material defects. *Materials Design*, 162:300–310, 2019. 16

[70] Anastacia MacAllister, Adam Kohl, and Eliot Winer. Using high-fidelity meta-models to improve performance of small dataset trained bayesian networks. *Expert Systems with Applications*, 139:112830, 2020. 16

[71] Yaqing Wang and Quanming Yao. Few-shot learning: A survey. *CoRR*, abs/1904.05046, 2019. 16

[72] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. Routledge, 2017. 23

[73] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 37

[74] Spyridon Trastelis. LSTM-based Autoencoders for Anomaly Detection in ASML Machines. unpublished, 2019. 44

[75] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *CoRR*, abs/1806.07366, 2018. 45

[76] Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. Latent odes for irregularly-sampled time series. *CoRR*, abs/1907.03907, 2019. 45

Appendix A

A.1 Selected features

These are the features selected by the tree-based models. The number of the features are the optimized numbers.

The 15 most important features for anomaly classification selected by decision tree from 234 total features

Z10_3_Totaldrift
Z21_2_Residualdrift
Z8_0_Totaldrift
Z7_0_Residualdrift
Z27_3_Residualdrift
Z11_0_Residualdrift
Z14_3_Totaldrift
Z3_1_Residualdrift
Z3_3_Residualdrift
Z12_2_Residualdrift
Z3_3_Totaldrift
Z17_0_Residualdrift
Z29_1_Residualdrift
Z2_0_Residualdrift
Z10_1_Totaldrift

The 14 most important features for anomaly classification selected by decision tree from 94 features

Z15_2_Totaldrift
Z20_1_Totaldrift
Z17_0_Residualdrift
Z9_2_Residualdrift
Z6_0_Residualdrift
Z20_1_Residualdrift
Z5_2_Totaldrift
Z10_3_Totaldrift
Z18_0_Totaldrift
Z27_1_Residualdrift
Z18_1_Residualdrift
Z17_0_Totaldrift
Z7_3_Totaldrift
Z20_0_Totaldrift

The 17 most important features for anomaly classification selected by xgboost from 234 features

Z20_1_Totaldrift
Z6_0_Residualdrift
Z10_3_Totaldrift
Z27_3_Residualdrift
Z15_2_Totaldrift
Z17_0_Residualdrift
Z25_1_Residualdrift
Z22_1_Residualdrift
Z21_0_Totaldrift
Z20_1_Residualdrift
Z14_2_Residualdrift
Z29_0_Residualdrift
Z9_2_Residualdrift
Z23_1_Residualdrift
Z34_3_Residualdrift
Z5_0_Totaldrift
Z30_3_Residualdrift

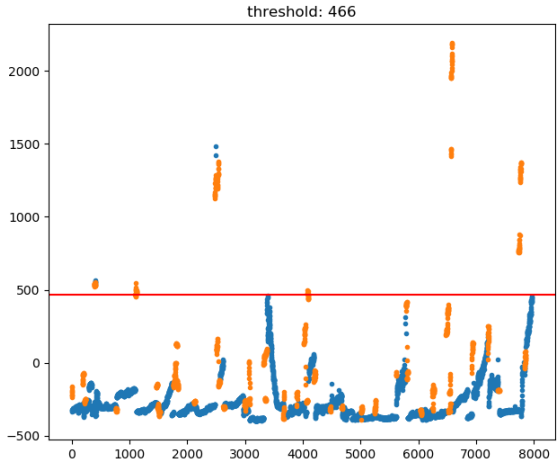
The 20 most important features for anomaly classification selected by xgboost from 94 features

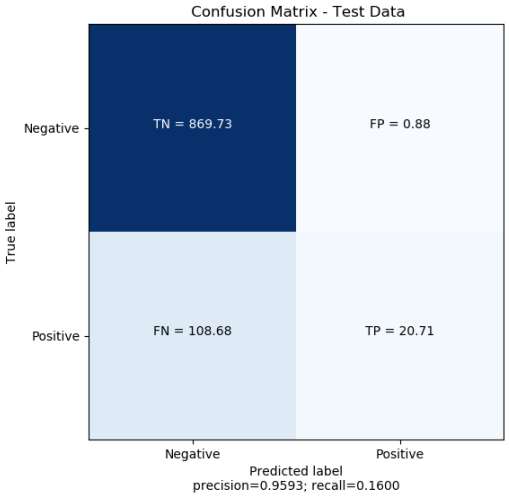
Z10_3_Totaldrift
Z20_1_Totaldrift
Z17_0_Residualdrift
Z20_1_Residualdrift
Z15_2_Totaldrift
Z6_0_Residualdrift
Z27_1_Residualdrift
Z5_1_Totaldrift
Z18_0_Totaldrift
Z7_3_Totaldrift
Z18_1_Residualdrift
Z9_2_Residualdrift
Z20_3_Totaldrift
Z15_3_Residualdrift
Z8_3_Residualdrift
Z8_1_Residualdrift
Z8_2_Residualdrift
Z11_0_Residualdrift
Z11_0_Totaldrift
Z24_0_Residualdrift

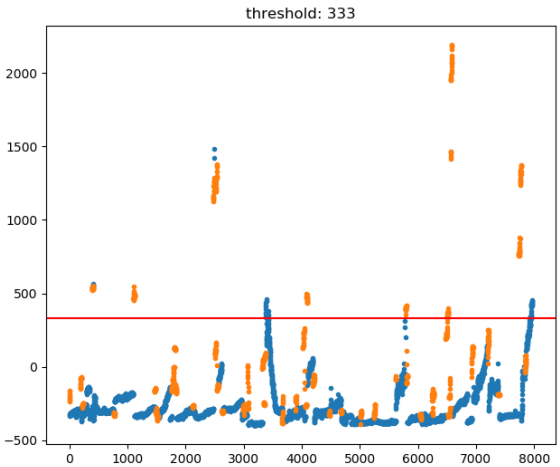
Appendix B

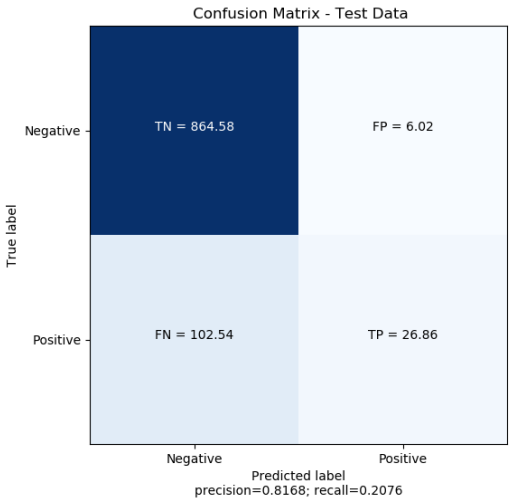
B.1 Confusion matrix under different thresholds

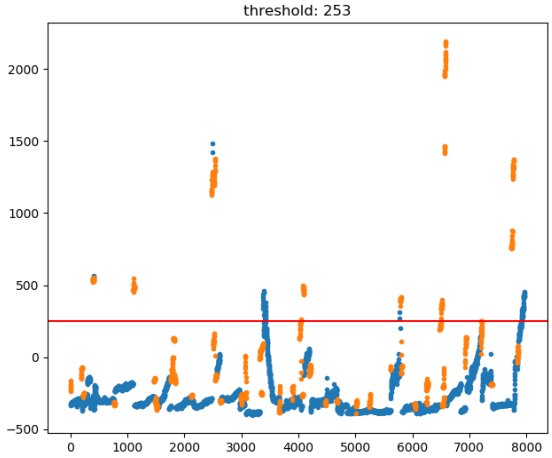
offline training with 234 zernike drifts features

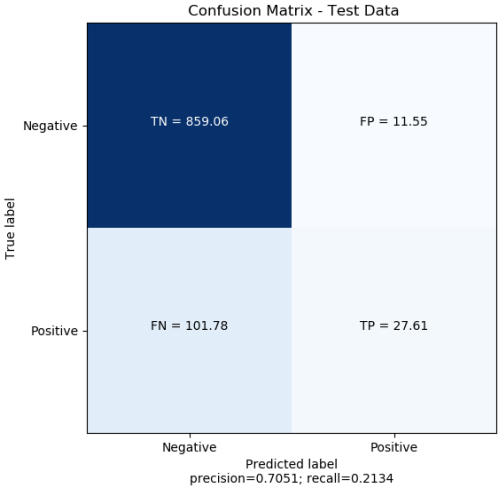


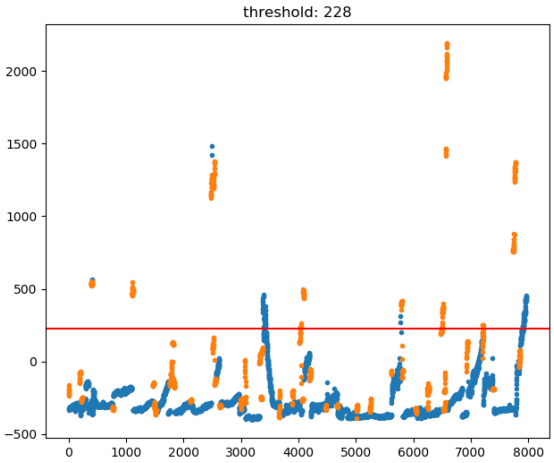


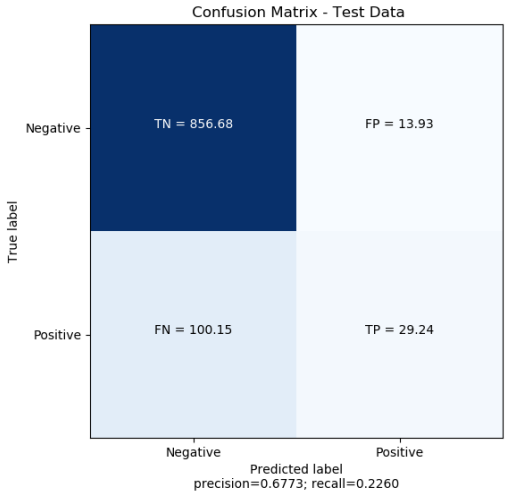





threshold: 253
2000
1500
1000
500
0
−500
0
1000
2000
3000
4000
5000
6000
7000
8000


Confusion Matrix - Test Data
Negative
Positive
True label
TN = 859.06
FP = 11.55
FN = 101.78
TP = 27.61
Negative
Positive
Predicted label
precision=0.7051; recall=0.2134


threshold: 228
2000
1500
1000
500
0
−500
0
1000
2000
3000
4000
5000
6000
7000
8000


Confusion Matrix - Test Data
Negative
Positive
True label
TN = 856.68
FP = 13.93
FN = 100.15
TP = 29.24
Negative
Positive
Predicted label
precision=0.6773; recall=0.2260

Appendix C

C.1 Dashboard

This appendix demonstrates the dashboard application developed by the author to dynamically visualize and evaluate the results of the detection model.

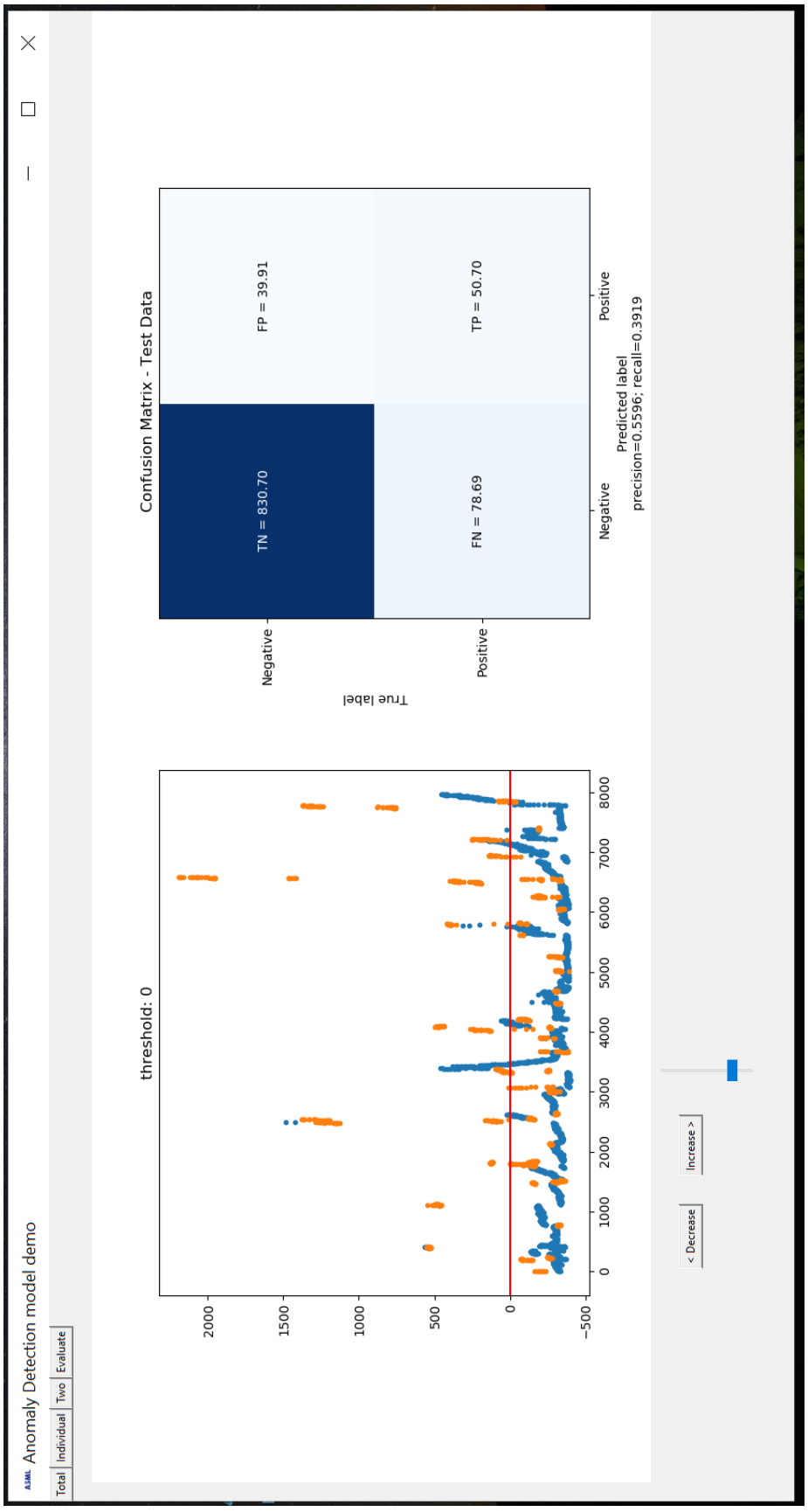


Figure C.1: Dashboard page for modifying the threshold manually

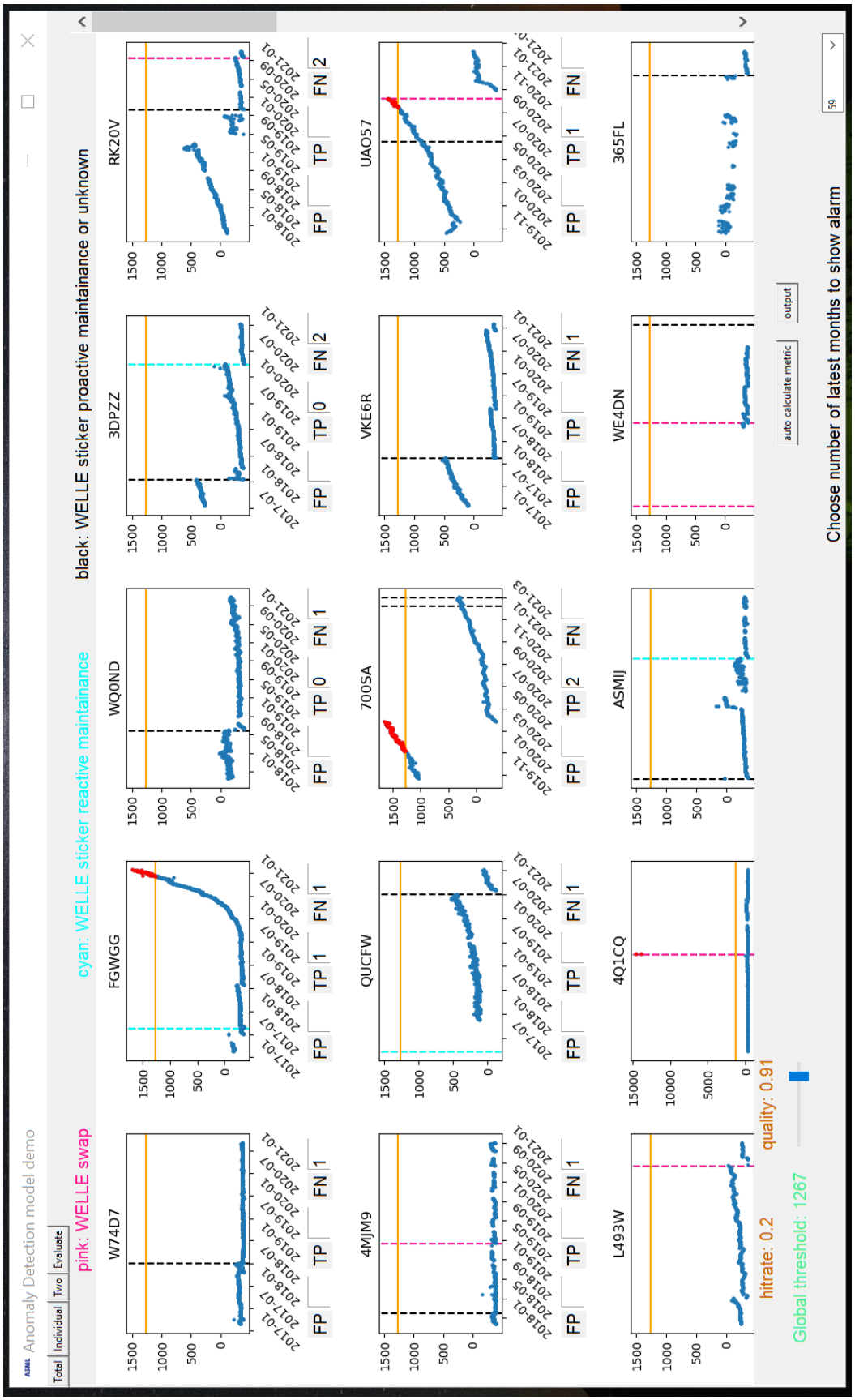


Figure C.2: Dashboard page for calculating hitrate and quality with manually input and automatically calculation functions (All machine names have been anonymized by using random strings

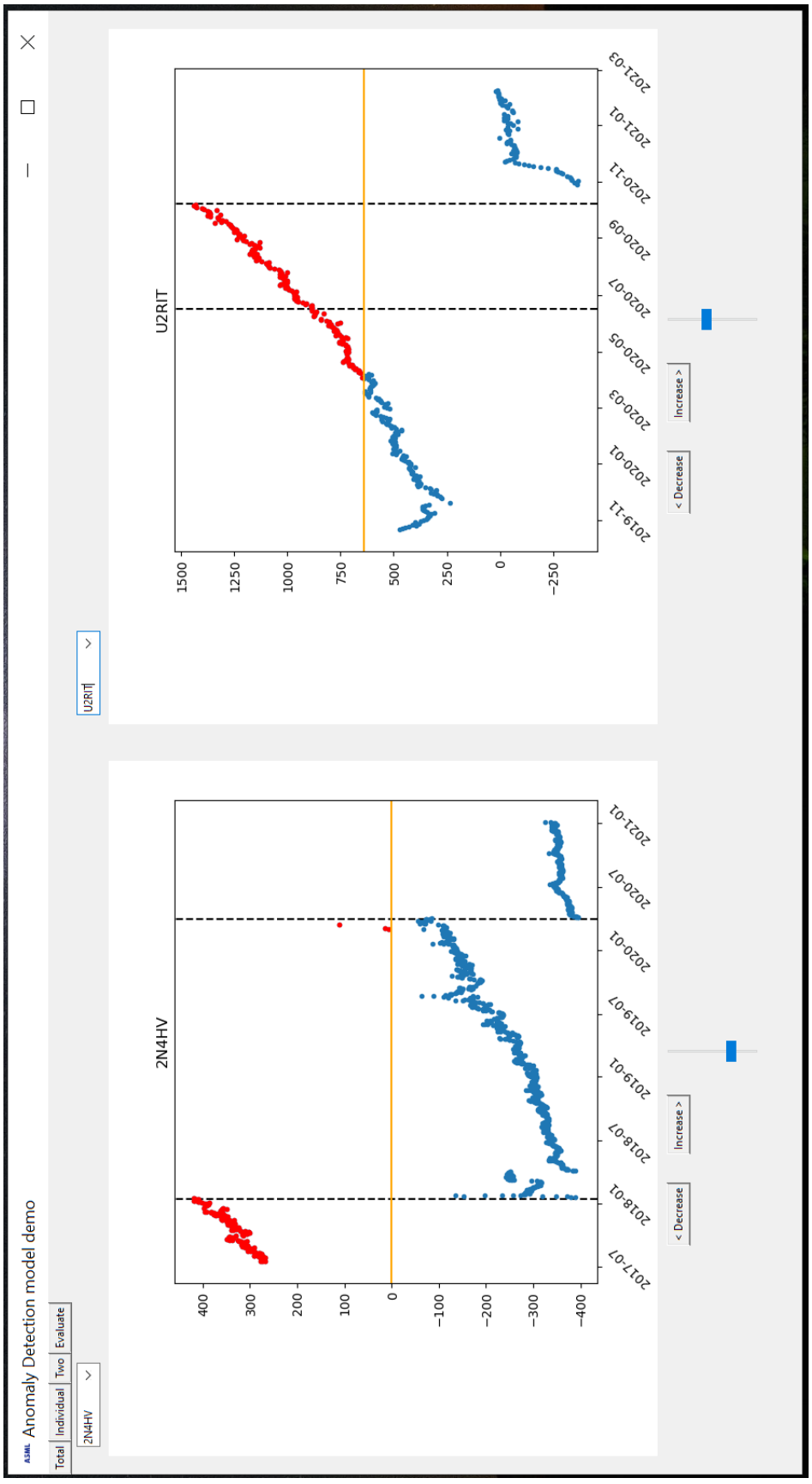


Figure C.3: Dashboard page for comparing 2 machines, machine names have been anonymized by using random string names