Unsupervised representation learning for time series anomaly detection and beyond

Ferreira Correia; L.

Chapter 7

Extending Representation Learning to Predictive Maintenance

7.1 Introduction

As demonstrated in Chapter 5.3, modelling nominal behaviour of dynamic systems allows for the detection of anomalies when no examples of anomalies are present. This procedure can also useful for other time series problems, such as predictive maintenance (PdM), which contrasts simpler maintenance strategies like run to failure (RTF) and preventative maintenance (PM). On the one hand, RTF involves running arbitrary dynamic systems until failure, and therefore leads to unexpected downtime, as well as higher costs and potential damage to other, otherwise healthy, components. On the other hand, PM involves the preemptive change of components in said arbitrary system before they break, also leading to higher costs, as components are changed despite having remaining useful life. PdM involves predicting the state of health of the dynamic system, which can then be used to inform a maintainer on the ideal time for maintenance, i.e. the time before a component within the system breaks but also has little remaining useful life.

Unlike anomaly detection, PdM does not try to separate data into two classes, as component deterioration is usually a gradual process. Modelling the nominal behaviour at the beginning of life of a dynamic system therefore allows for the quantifi-

cation of the deterioration level by comparing the model behaviour with the observed behaviour, since a real-world system experiences wear, whereas the model does not.

To illustrate the applicability of a data-driven approach for time series anomaly detection, like the temporal VAE (TeVAE), it is investigated based on a real-world case study. This real-world problem involves an electric motor (EM) on a test bench, with which a single drive cycle is driven repeatedly. Similar to the powertrain introduced in Chapter 4.2, the EM is connected to two resistances which simulate the real-world road load. This EM consists of a stator and a rotor, which are both cooled and lubricated by an oil-filled loop. Due to the rotating nature of the rotor, a special manifold is used to distribute the oil, which, in this case study, twists with time due to human error during installation. The twisting motion is very gradual but leads to a reduction of the cross-sectional area the oil can flow through, impacting the cooling capacity available to the rotor. This would usually manifest itself in features like the rotor temperature, though, given the rotating nature, no sensors are present to quantify such properties. Oil pressure, in contrast, is monitored by a simple rule-based system, and it could be used to detect the reduction of cross-sectional area, though this is a quantity that would not typically be relevant unless this type of failure is anticipated. Considering this, as well as the absence of a physical metric that can serve as a health indicator, and the lack of labelled failure data due to twisted oil distributors, this is classified as an unsupervised PdM problem [116, 117], which can be approached by computing a virtual health indicator in form of the anomaly score output by TeVAE. Unlike a physical health indicator, though, a virtual health indicator can no longer be interpreted as the quantification of a physical condition.

Additionally, this EM has an integrated two-stage gearbox, which decouples its angular speed from the axle angular speed. Earlier chapters in this thesis did not have to consider such a change in discrete states, hence TeVAE is modified to regard the current gear signal as an exogenous input feature which is not reconstructed and therefore does not contribute to the health indicator.

7.2 Experimental Setup

The features selected to represent testee behaviour are once again based on domain knowledge, and somewhat follow the choices made in Chapter 4. They are shown in tabulated form in Table 7.1. Additionally, Figure 7.1 shows an example of a test sequence plotted for each of the channels.

The available dataset $\mathcal{D}$ consists of 1048 tests, which after an applied training and

Table 7.1: Channels (features) chosen for modelling testee behaviour.

Index	Signal Name	Index	Signal Name
1	Vehicle Speed	7	Inverter Temperature
2	EM Voltage	8	Left Axle Torque
3	EM Current	9	Right Axle Torque
4	EM Torque	10	Left Axle Angular Speed
5	EM Angular Speed	11	Right Axle Angular Speed
6	EM Stator Temperature	12	Current Gear

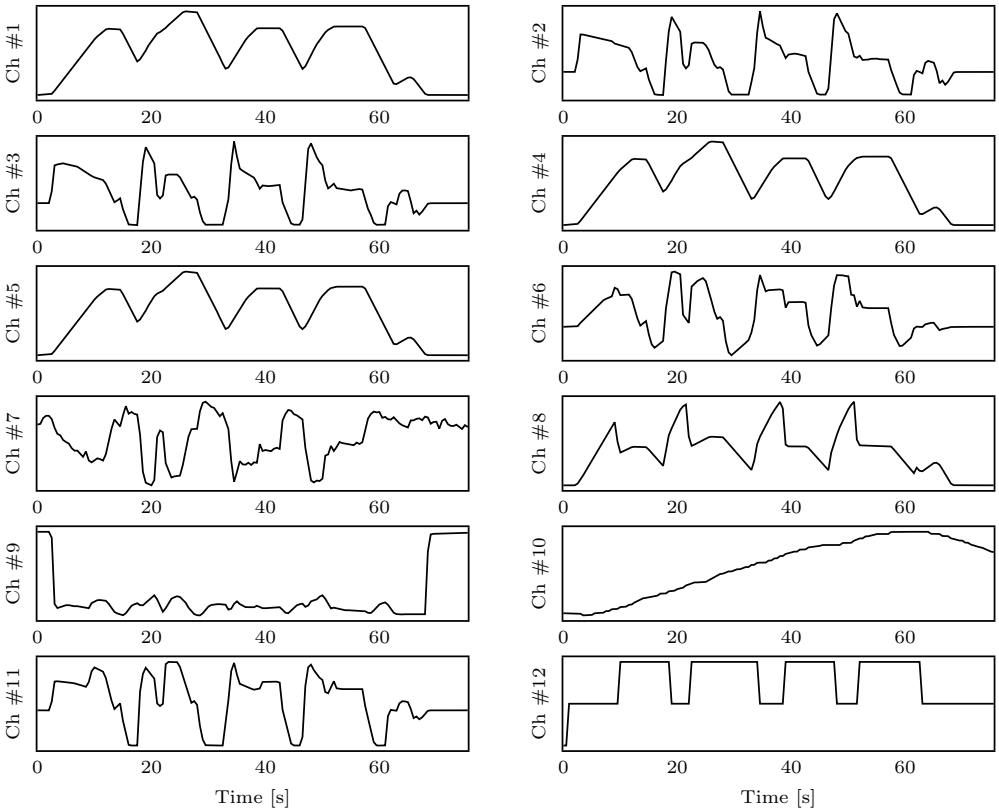

Figure 7.1: Sample plot of a test sequence. The channel legend can be found in Table 7.1. The y-axis ticks are removed due to comply with confidentiality requirements.

test split of 1:2 corresponds to a training subset $\mathcal{D}^{\text{train}}$ of size $M = 349$ and a test subset $\mathcal{D}^{\text{test}}$ of size $N = 699$. It should be noted that, unlike previous experiments, $\mathcal{D}$ is not shuffled before being split to maintain the temporal order of tests. Like in Chapter 5.2, 20 % of the unlabelled training subset $\mathcal{D}^{\text{train}}$ is separated to form the validation subset

$\mathcal{D}^{\text{val}}$, later used for early stopping and threshold estimation. Following the procedure from Chapter 4, the data is low-pass-filtered and downsampled to a sampling frequency of 2 Hz and then windowed to a window size of $w = 16$ time steps. As is evident, the window size is much smaller compared to the datasets discussed in Chapter 4, which stems from the fact that, unlike the others, this dynamic system lacks a battery with the relatively slow temperature and state of charge signals.

TeVAE is used as the underlying model and is trained as specified in Chapter 5.2. The framework used for model training is TensorFlow 2.15.1 and TensorFlow Probability 0.23 on Python 3.10 on a workstation running Ubuntu 22.04.5 LTS, equipped with two Nvidia RTX A6000 GPUs. Further information on library versions used can be found in the *requirements.txt* file in the GitHub repository under `github.com/lcs-crr/Thesis`.

7.3 Results

When TeVAE is applied to the test subset $\mathcal{D}^{\text{test}}$, a health indicator as a function of time for each sequence is generated. To illustrate the development of the health indicator with time, the maximum value of each sequence is taken and plotted against the test sequence index in Figure 7.2.

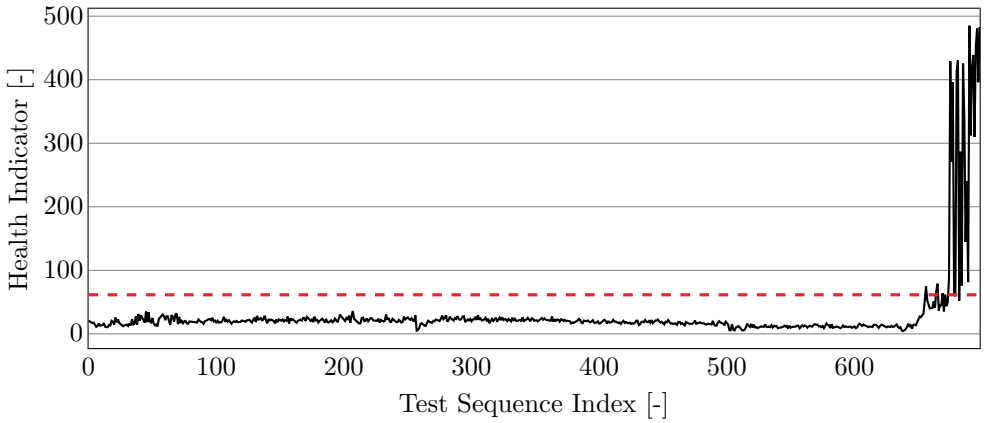

Figure 7.2: Health indicator plotted as a function of the test sequence index. The red dashed line represents the unsupervised threshold obtained using the health indices within the validation subset.

As is evident, the health indicator remains largely constant for most of the test set. At test sequence index 656, the health indicator first crosses the threshold, which

serves as an indicator for users that the model behaviour is significantly deviating from the observed behaviour and that the system or a component within it is starting to deteriorate. The rule-based oil pressure monitor only warned the user at the last test sequence and, considering that each test sequence is around 75 s long, this corresponds to an improvement in deterioration detection by around 54 min.

Considering that TeVAE is essentially used in the same way as it was in Chapter 5.2, it can serve as an approach for both time series anomaly detection and predictive maintenance. During inference, once a sequence is input and reconstructed by TeVAE, it can be used for both problems at no additional computational cost, setting a low hurdle for such a dual operation in the real world.

7.4 Conclusions

This chapter shows that, in the case of a real-world predictive maintenance problem, TeVAE can quantify the change in system behaviour due to deterioration. This is especially remarkable considering that the two-fold unsupervised nature of the problem, which encompasses the lack of a physical and observable health indicator, as well as the absence of labelled historical data depicting failure due to a twisted oil distributor.

Future work should revolve around providing a more sophisticated, perhaps component-level, health indicator, as currently the health indicator only represents the health of the system as a whole. A component-level health indicator allows for more precise diagnostics of a potential problem, especially with very complex systems consisting of several subsystems.

Chapter 8

Conclusions and Outlook

This thesis represents a summary of the research done in the field of time series anomaly detection. As identified in Chapter 3, several gaps in the literature are identified and formulated in the form of research questions in Chapter 1.2. This thesis addresses these by contributing to the aspect of benchmarking, robust modelling, as well as intelligent querying of labels to inform a superior threshold choice. In this chapter, the research questions are restated, and the corresponding answers briefly summarised.

RQ1 Can a non-trivial multivariate time series dataset be generated from physically motivated simulation models?

As shown in Chapter 4, it is indeed feasible to generate a synthetic dataset using simulation tools. Thanks to the controlled environment enabled by simulation, almost full control is gained over the generation procedure. As is typical, simulation allows for the possibility of faster-than-real-time computation, while also offering much more flexibility. This is especially evident in the case of the variable initial states like battery temperature and state of charge used in the powertrain anomaly time series benchmark (PATH) dataset. In the real world, this would require laborious charging/discharging and heating/cooling of the battery before any drive cycle, while in the case of a simulation only two scalar parameters need to be adjusted. For anomaly detection specifically, using a simulation model is associated with a unique set of benefits too. Recall Chapter 3, where the correctness of ground-truth label vectors for some publicly available datasets are questioned. A simulation model, in contrast, allows for precise timing for the start of anomalies and therefore ground-truth labels. It should

be noted, however, that even in simulation it can be difficult to pinpoint the end of an anomaly in the resulting data, as the anomalous behaviour may still be present but not observable due to slow dynamics of some internal processes, for example. With a simulation model, anomalies can also be generated with less bias and more realistically than manually tampering with the data, by provoking anomalous behaviour using, for instance, the change of a parameter before simulation. Lastly, a simulation model can, with some domain expertise, be extended to increase complexity and therefore generate data that is even closer to the real world. While the PATH dataset is based on a simulation model from the automotive domain, other simulation models from other domains could also be used to create a portfolio of different datasets, which would enrich the benchmarking aspect of the anomaly detection research field.

RQ2 Are satisfactory results obtainable when detecting anomalous behaviour in multivariate time series using completely unsupervised methods?

On the one hand, the answer to this research question may seem straightforward going by the results for the unsupervised version of the PATH dataset provided in Chapter 5.3. Recall that this version features a contaminated training subset with around 8 % unlabelled anomalous sequences and hence not only poses a challenge in terms of robust modelling of nominal behaviour but also in terms of threshold setting. Especially the latter problem manifests itself clearly, as truly unsupervised approaches, i.e. trained on unlabelled data and using the unsupervised threshold are far outmatched by their counterparts trained on unlabelled data but calibrated using the best possible threshold. It becomes clear that the unsupervised threshold is the weakest link in the anomaly detection algorithm, as it is highly sensitive to unlabelled anomalous data. But what if the training subset is not contaminated? These results further underline the high sensitivity of the threshold, as without contamination the results obtained using the unsupervised threshold exceed the previous ones by a large margin and feature a much smaller gap to their counterpart obtained using the best possible threshold. The results using the real-world dataset, which is also free of anomalies in the training subset, further contribute to this observation. Approaches fitted on a training subset with nominal only data, however, are classified as semi-supervised, and hence satisfactory results are not obtainable using truly unsupervised methods, highlighting the need for subsequent research on threshold choice in such cases.

RQ3 How well can a time series anomaly detection approach generalise to unseen data stemming from the same dynamic system?

To answer this research question, the PATH dataset is split into different cases where each is missing sequences with either low, medium or high average vehicle speeds or dynamism levels. A model that learns the underlying nominal behaviour of the dynamic system should in theory be able to generalise and represent unseen behaviour, as long as it is not in an extreme operational range. This can be a challenge, as the approach must be able to discern unseen but nominal behaviour from anomalous behaviour. The results show that, the absence of highly dynamic time series data and high average vehicle speed takes an especially large toll on the anomaly detection results, while generally, the opposite is true for low average speed and low dynamism data.

RQ4 Can active learning be harnessed to find a suitable threshold for completely unsupervised methods?

Recall the answer to RQ2. Considering the high sensitivity of the unsupervised threshold when faced with contaminated and unlabelled datasets, there is a clear need for a more robust threshold estimation method. Active learning can provide alleviation to this problem by intelligently querying a domain expert and estimating a threshold based on the labelled data. Clearly, such a procedure is no longer truly unsupervised, though that does not mean that it cannot find application in the real world. Furthermore, countless publications in the literature simply assume labelled data to be available when setting hyperparameters and thresholds, as is discussed in Chapter 3. In contrast, using active learning for setting a threshold forgoes this assumption and experiments associated attempt to simulate this challenge. Results show that, obtaining a threshold using active learning outperforms the unsupervised threshold by a large margin, depending on the query strategy employed. Compared to more traditional query strategies, the novel dissimilarity-based querying strategy outperforms in almost all cases, with particularly large margins in very low querying budget scenarios.

RQ5 How robust are active learning-based thresholds to mislabelling by the oracle?

The term of a domain expert is fairly loosely defined. Generally, it is interpreted as someone who is well versed in their application, though this can be at different levels depending on the individual. Furthermore, humans make mistakes and hence the impact of mislabelling on anomaly detection performance is investigated. This is especially interesting for practitioners to gauge how robust the estimation of thresholds using active learning can be in the real world. The experiments undertaken demonstrate that most of the results for all query strategies remain above the unsupervised

threshold, representing an improvement, even at fairly high mislabelling rates of three in ten, for instance. While promising, the results could still be improved to further approach those obtained using the best possible threshold.

Beyond the future work resulting from the research questions, further research should be dedicated to other aspects. These relate to benchmarking only, as it sets the foundation for measurable advancements in the future. While a set of metrics apt for online evaluation of discrete-sequence problems is proposed in this work, they still require further development, as outlined in Chapter 5. An adaptation of the improved range-based metrics [33] to discrete-sequence problems in addition to continuous-sequence problems could serve as a solution. In addition to that, approach complexity needs to be quantified as normalised for when presenting results. In the field of optimisation, this is often done by denoting performance in relation to function calls, for example. While the complexity of parametric approaches can be loosely described by their parameter count, two different architectures with the same number of parameters may have two entirely different runtimes on the same hardware or benefit from acceleration from certain hardware. Furthermore, the parameter count cannot be obtained for non-parametric approaches, which makes the comparison between a non-zero parameter approach with a non-parametric approach difficult. Once a strong foundation is set for benchmarking and associated procedures are widely agreed upon, advances benefiting the fields of medical field [1] and server machines [2] to water distribution [3] and treatment [4] can be achieved.