Artificial Intelligence for Engineering Design, Analysis and Manufacturing

www.cambridge.org/aie

A semi-supervised anomaly detection approach for detecting mechanical failures

Colin Soete, Michaël Rademaker and Sofie Van Hoecke

IDLab, Ghent University, IMEC, Technologiepark-Zwijnaarde 126, Ghent, 9052, Belgium

Research Article

Cite this article: Soete C, Rademaker M and Van Hoecke S (2024). A semi-supervised anomaly detection approach for detecting mechanical failures. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **38**, e16, 1–12
https://doi.org/10.1017/S0890060424000131

Received: 19 September 2023
Revised: 21 May 2024
Accepted: 09 June 2024

Keywords:
labeling; predictive maintenance; semi-supervised learning; anomaly detection

Corresponding author:
Colin Soete;
Email: Colin.Soete@UGent.be

Abstract

Predictive maintenance attempts to prevent unscheduled downtime by scheduling maintenance before expected failures and/or breakdowns while maximally optimizing uptime. However, this is a non-trivial problem, which requires sufficient data analytics knowledge and labeled data, either to design supervised fault detection models or to evaluate the performance of unsupervised models. While today most companies collect data by adding sensors to their machinery, the majority of this data is unfortunately not labeled. Moreover, labeling requires expert knowledge and is very cumbersome. To solve this mismatch, we present an architecture that guides experts, only requiring them to label a very small subset of the data compared to today's standard labeling campaigns that are used when designing predictive maintenance solutions. We use auto-encoders to highlight potential anomalies and clustering approaches to group these anomalies into (potential) failure types. The accompanied dashboard then presents the anomalies to domain experts for labeling. In this way, we enable domain experts to enrich routinely collected machine data with business intelligence via a user-friendly hybrid model, combining auto-encoder models with labeling steps and supervised models. Ultimately, the labeled failure data allows for creating better failure prediction models, which in turn enables more effective predictive maintenance. More specifically, our architecture gets rid of cumbersome labeling tasks, allowing companies to make maximum use of their data and expert knowledge to ultimately increase their profit. Using our methodology, we achieve a labeling gain of 90% at best compared to standard labeling tasks.

Introduction

Industrial workflow optimization typically requires high reliability of mechanical equipment to avoid bottlenecks or overcapacity. When mechanical equipment breaks down, both the repair cost and the opportunity cost increase due to reduced capacity. The tradeoff between maintenance and breakdown costs gives rise to three different maintenance strategies; the corresponding impact on costs is shown in Figure 1. First, *corrective/reactive maintenance* (CM) is where the machine is used to its limits, and repairs are performed only after a machine failure. Second, *preventive maintenance* (PM) is where machines are routinely serviced without taking their operating characteristics into account. When scheduling service maintenance, planning is often too conservative, resulting in sub-optimal machine availability. And third, *predictive maintenance* (PdM) is where maintenance is scheduled only when needed, close to a breakdown in the machine or machine efficiency, by dynamically predicting this machine breakdown. In the latter case, the costs of maintenance and breakdown are combined to determine an optimal tradeoff.

Thanks to the rise of the Industrial Internet of Things (IIoT), many companies started collecting data to monitor their assets (Yan et al. 2017). However, when PwC and Mainnovation (Mulders and Haarman 2017) audited up to 290 Belgian, Dutch, and German companies about their current use of predictive maintenance, they found that around 70% of the companies still schedule maintenance based solely on periodic inspections and conclusions revolving around a combination of the inspector's expertise and instrument readouts. Only 11% of the surveyed companies perform continuous real-time monitoring of assets and environmental data with alerts based on predictive techniques such as regression analysis. A significant portion of the companies, about 30%, said that the lack of sufficient labeled data (or lack of data analytics capabilities) is the main reason for not applying predictive maintenance.

Companies today do have a substantial amount of (unlabeled) data available; however, the process of labeling data is too tedious, and requires domain experts, a scarce and costly profile, to repeatedly label similar data points. To exploit the power of this data, one should therefore find solutions that facilitate labeling the data by maximally using the domain expert's knowledge in as little time as possible, so as to gather the labeled data needed to shift toward predictive maintenance.

To alleviate the need for labeled data, today, extensive research is done on unsupervised anomaly detection (AD) techniques. However, also in this field, the lack of (labeled) ground truth

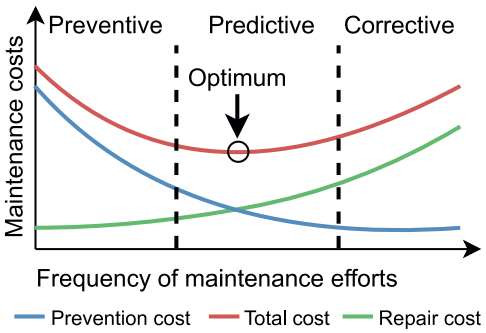

Figure 1. Differences in maintenance costs regarding the various strategies of performing maintenance, such as preventive maintenance, predictive maintenance, and corrective maintenance.

data makes the evaluation of these unsupervised architectures troublesome, resulting in a lack of trust in the ensuing models.

Therefore, to address the need for labeled data without bothering domain experts with cumbersome labeling tasks, this paper presents a framework that guides experts so they only have to label a very small subset of data compared to today's standard labeling campaigns when designing predictive maintenance solutions. To do so, our presented framework aims to provide insights into the raw data of machinery with the detection of anomalies as a result. Next, these anomalies are enriched with business intelligence to make the complete architecture more robust and efficient. This leads to several positive opportunities. The first and biggest one is the automated identification of a repetitive or similar failure, which, when acted upon, can be prevented. The second opportunity is decision-making based on the data while still taking experts' feedback into account. Another opportunity is the identification of harmful operational parameters. For example, a temperature parameter during a process could spike beyond its designated bounds, resulting in damage to the equipment. Finally, compared to the current state of the art (SOTA), our solution proposes this architecture without requiring larger labeling tasks.

Our proposed architecture also enables today's unsupervised solutions, with small adjustments, to be transformed into semi-supervised systems. This transformation will lead to a higher certainty and trust in the detection of anomalies since these anomalies are verified by an expert. The presented solution also focuses heavily on explainability as this is crucial in business sectors, such as the pharmaceutical and healthcare industries, to gain trust in the designed models. After all, an inaccurate interpretation of a single model can lead to disastrous consequences for valuable assets. Therefore, multiple techniques are employed to enhance the interpretability of these models. By involving expert knowledge at the base of these models, different fault types (FT) can be pinpointed and explained.

The remainder of this paper is as follows. In Section Related work the related work and shortcomings of today's approaches are discussed. Section Use case describes the use case of this paper and the specifications of the data. The extraction process from raw data to features is also explained. In Section Architecture, the architecture and different components that compose the entire pipeline are presented, i.e. the auto-encoder models to flag anomalies, the clustering of flagged anomalies, and the supervised model that checks the flagged anomalies with experts' feedback. Also, throughout Section Architecture, the results of each component are discussed and evaluated. Section General results describe the general results of the full architecture. Finally, Section Conclusions draws conclusions that we gained from the research and discusses the potential future work.

Related work

This paper is the first to present an entire pipeline to minimize the labeling effort needed when designing predictive maintenance solutions, but parts of the pipeline resemble other works in the literature, which are discussed here. These approaches however mostly lack the connection with labeling data to optimize predictive maintenance. The next paragraphs list those approaches that are, in part, similar to ours and list their pitfalls along with the solutions provided by our research.

It is essential to preface that due to the nature of our labels, supervised methods for distinguishing anomalous data from normal data cannot be utilized. Consequently, approaches such as one-vs-the-rest multiclass strategies are not applicable to this research as these are supervised and also require labels of unhealthy data. The specifics of our labeling criteria will be elucidated in subsection auto-encoder models.

A wide range of *AD* techniques are currently utilized in a predictive maintenance setting. The Matrix Profile (Lu et al. 2023; De Paepe et al. 2020) is a heavily-supported time series analysis technique that can be used for motif discovery, AD, segmentation, and others. Other approaches were however preferred here over a matrix profile approach as we are only certain about the origins of our healthy data, and thus matrix profile, more efficient to use on (faulty) discords, suffered from a too high time complexity. However, a matrix profile is shown to be an effective approach when zooming in on one particular anomaly and examining the nearest matches of that anomaly. Hence, matrix profiles could be incorporated as a tool to enhance explainability by finding correlated (earlier) motifs that might be the root causes of the anomaly.

Another popular approach for AD is one-class classification-based models, which focus on learning a description of a set of data instances to determine whether new instances adhere to the training data or not. Most one-class classification models are inspired by the support vector machine (SVM), including two popular variants: the one-class SVM and the support vector data description. However, these one-class models may work ineffectively in datasets characterized by complex distributions within the normal class (Hindy et al. 2020).

Deep learning approaches are also frequently used for AD and benefit from the fact that they can handle an overload of data in a relatively short amount of time (Pang et al. 2021). Nasir and Sassani (2021) show that deep learning methods, such as convolutional neural networks and deep belief networks are frequently used to solve problems related to Industry 4.0. The same research declares that auto-encoders (AE) are a popular technique to track down anomalous behavior within this recent industry standard. Research by Li et al. (2019) uses a similar approach to the first part of our framework, i.e. the preprocessing and auto-encoder steps. They use stacked AE following a long short-term memory (LSTM) architecture. Furthermore, Fährmann et al. (2022) leverage LSTM variational AE for AD in industrial control systems. These systems are susceptible to intrusion through manipulation of sensor or actuator data, resulting in anomalous behavior. Their specific type of AE is evaluated for detecting sensor and actuator faults in water treatment facilities.

Their and other research using AE as an AD technique attempt to solve the lack of unlabeled data with this unsupervised learning approach. However, as far as we know, AE have not been used as a tool to enhance labeling tasks in a predictive maintenance context.

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

More complex deep learning AD techniques, like generative adversarial networks, often suffer from convergence issues, particularly when dealing with complex true data distributions or unexpected outliers in the training data. In contrast, AE offer a straightforward and versatile approach applicable to various types of data. Additionally, anomaly-measure-dependent feature learning approaches were also disregarded as these techniques aim at learning feature representations that are specifically optimized for only one particular existing anomaly measure.

Clustering methods can be used to divide data into groups of failure types (Serradilla et al. 2021). A practical example by Gittler et al. (2021) utilizes a multivariate dataset with vibration and acoustic emission signals to predict the flank wear of milling machinery. The signals are segmented into bins of a fixed window length of which features are extracted and then filtered. Several samples are taken during degradation and are then clustered by their most relevant features. Visualization using t-SNE is then employed to compare the clusters with actual flank wear and see if they correspond with each other. One of the main shortcomings of this and similar approaches is that they only focus on one fault type to be evaluated, and assume that the data can be perfectly clustered and mapped without overlap of other (latent) fault types. Li, Li, and Ma (2020) have a similar approach to our pipeline, utilizing an auto-encoder to compress machinery data into a latent space, followed by k-means clustering to distinguish faults. However, clustering directly on high-dimensional data poses challenges, particularly in determining the appropriate number of clusters in advance. Moreover, clustering anomalies directly from the training data (Li et al. 2021; Pu et al. 2020) would therefore not be feasible for such high-dimensional data in our specific use case.

A review by Weichert et al. (2019) delves into the current shortcomings of machine learning (ML) algorithms employed to optimize manufacturing production processes. They highlight a prevalent issue: the disconnect between process complexity, data volume, and model complexity. This disconnect often leads to overfitting and a lack of interpretability, contributing to skepticism regarding the applicability of ML in the manufacturing sector (Chakraborty et al. 2017; Li et al. 2022). In response to these challenges, our methodology enhances the interpretability of deep learning models. We achieve this by labeling only a small subset of anomalies, allowing these models to be effectively deployed across various manufacturing industries.

In conclusion, the main drawbacks, such as the lack of ground truth, clustering issues, and interpretability of current related work need to be addressed and evaluated in a real-life use case. The benefits of present promising work need to be exploited by compiling it into a logical pipeline while facilitating efficient labeling.

Use case

An appropriate example use case where a lack of labeled data is prevalent and where maintenance costs escalate is the pharmaceutical industry. Mainly for this industry, having a wide variety of mobile or fixed assets is a hurdle to performing predictive maintenance. Also, within this industry, the standard for calculating equipment reliability is very complex when assets are not used 100% of the time or when they are multifunctional. The selected use case resembles a real-life situation where assets experience reliability performance issues.

The dataset contains a multivariate time series per machine. The investigated machines, several dryers, and filters are listed in Table 1. Note that this research is not limited to the presented use case and its specific machines but can also be deployed on other machines and/or use cases. The data in our setup are impacted by several different (latent) fault types which are mostly unique for each machine. The appliances are equipped with sensors that read out information such as temperature, pressure, and valve statuses. Each signal of an asset is currently fetched from a database with a sample rate of at least one sample per minute. Signals with a higher sample rate are down-sampled to one sample per second as this rate is high enough to capture all the signals' relevant details.

Machine learning models benefit greatly when given qualitative data, as irrelevant data lowers the performance of these models. Hence, it can be beneficial to clean and standardize the data before performing analytical analyses. The time series are cleaned by removing rows containing NaNs. By discarding these NaNs, the data are left with gaps in the time series that could be estimated. However, estimation could cause problems related to fabricated data because the data is generated through a model instead of gathering actual observations. Generating data through a model can have problems regarding extrapolation, missing certain spikes, or even making false assumptions. To avoid problems caused by imputation, a specific kind of sliding window is designed. A sliding window has two important characteristics: the window size and the stride size. The window size signifies the length of the window to aggregate over in minutes. The stride size denotes the amount of time between two consecutive windows. Usually, both the window size and the stride size are defined as a positive integer corresponding to a number of samples or measurements. However, we propose to use a datetime datatype for both sizes instead of a fixed integer.

Table 1. Specifications of the machines and their data

Machine	Time span	# Datapoints	# Signals	# Outliers	# Inspections[a]	# Rolling windows[b]	
						240_60	480_60
D001	731 days	1,042,246	17	22	2	6757	8,493
D009	1096 days	1,553,775	17	115	9	8425	10,288
D020	579 days	823,175	17	91	1	2536	2,998
F005	921 days	1,310,628	21	0	19	13,937	14,890
F008	943 days	1,324,788	21	0	14	12,549	13,770
F012	943 days	1,330,158	21	2	20	11,416	13,256
F017	921 days	1,315,540	21	6	28	12,519	13645

[a]The number of inspections refers to the manual inspections from technicians who intervene when any kind of error has occurred.
[b]The first integer indicates the window size in minutes, while the second integer specifies the stride size of the window.

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

Basically, this means that the proposed method consists of a variable number of samples per window instead of a fixed amount per window, and thus statistics are calculated in a time-based way. The determination of which window and stride sizes should be selected is based on the specific use case. For our use case, the window size was chosen based on the 4- and 8-hour shifts that the machines were operating on as seen in the final two columns of Table 1. However, variable window sizes could also have been a viable alternative. For example, these window sizes could then be defined using the start and end timestamps of a production process.

Columns containing NaNs or signals that carry no valuable information that helps with the prediction of faults are also discarded. In addition, outliers are erased before the sliding window procedure based on the physical constraints of each machine. Those particular samples containing signal values that exceed the physical range of the according sensor are thus omitted. As an example, values of 99,999 °C are not feasible in temperature signals where an interval of 0°C to 100°C is expected. This behavior could be induced by sensor interfacing errors leading to infeasible values.

Feature extraction is then performed on the cleaned windows with the tsfresh library (Christ et al. 2018) to obtain satisfactory features. These features are constructed to contain sufficient information to pinpoint mechanical failures and are thus selected together with domain experts. The selected features are limited in size to restrict the high computational power needed to extract these features and can be found under the Appendix Selected feature subset list. The features are then subjected to a filter which omits all records for when the machinery is non-operational. The importance of using this filter relies on the assumption that a mechanical failure is unlikely to occur when a machine is not in operating mode.

Architecture

The next three subsections outline the entire pipeline, from ingesting the features to automatically flagging recurring failures. Figure 2 provides a high-level view of the flow of the pipeline. In the following paragraphs, the first subsection, Section Auto-encoder models, discusses the AE models fed by the features mentioned in *Section Use case*. The obtained anomalies from these models are then evaluated and scrutinized. *Section Anomalies clustering* elaborates on clustering the flagged anomalies into groups with similar failure fingerprints. The performance of the clustering techniques is examined in the latter part of this subsection. The last subsection, Section Supervised system, discusses the supervised system that uses business intelligence to create a more robust and resilient hybrid model that is able to automatically detect seen failure types. The final part focuses on the metrics gained from these supervised systems.

Auto-encoder models

After filtering the features, a split of the feature vectors into healthy and uncertain parts is made. Failures progress over time and early signs of these failures arise days before the actual fault becomes visible. Hence, the estimation of when the failure likely originated, which also depends on the machine, is performed by an expert. The healthy data should reflect the machinery at its normal operation mode and thus should not contain any failures, as this would lead to a contaminated dataset when training an auto-encoder. The expert therefore ensures that the healthy data contains no faults by putting the split in a range when the fault could not have originated. Consequently, the rest of the data are labeled as uncertain and correspond to the days leading up to a manual inspection of a failure, likely having healthy data mixed with faulty data. For our use case, experts made this decision based on PF (Potential Failure) curves (Ochella, Shafiee, and Sansom 2021). By examining the potential faults for each machine within this curve, the split into healthy and uncertain parts can be achieved.

Next, the healthy segment is further split up into a train and validation set. A model per machine is trained on the first 80% of the data, while the model is validated on the last 20% of the data. No generic model and cross-validation amongst machines can be used as each machine has different specifications, resulting in the need for machine-specific models. By splitting the data per machine without shuffling, temporal data leakage is avoided when validating the model.

The selected model, an AE, is a form of neural network that has two main parts, namely the encoder and the decoder. The encoder maps the input, thus the features, into the code layer, which is usually the middle layer. This middle layer has fewer nodes than the input and output layers. The data is thus compressed into a smaller, latent space during the encoding phase. This phenomenon is actually fruitful as it acts as a feature selection method. The decoder then maps the code layer back to a reconstruction of the input. As the input and output layers have the same size, AEs are commonly symmetrical. In short, AE models try to reconstruct their input as accurately as possible using the compressed, latent, middle layer.

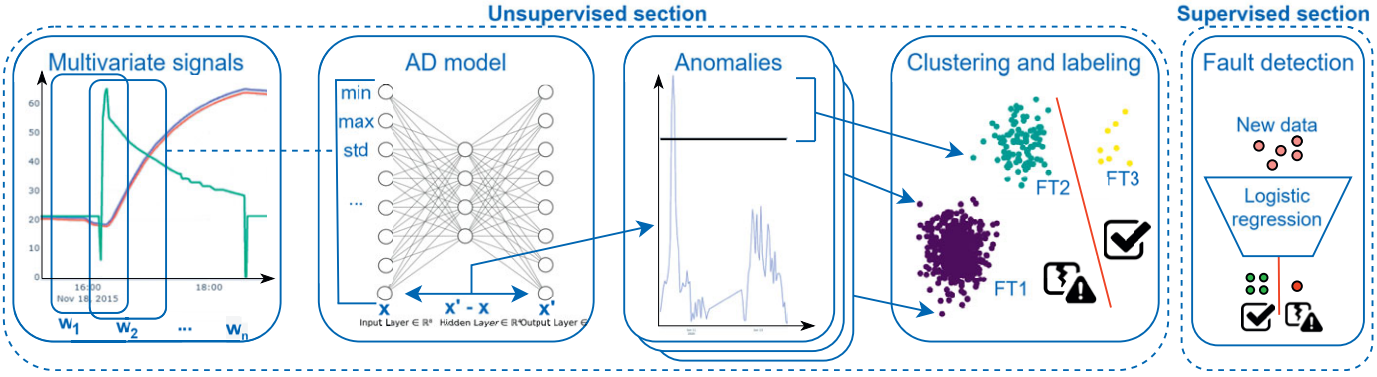

Figure 2. Overview of the architecture. The features of the multivariate signals are fetched into the AD models which return anomalies. These anomalies are then clustered according to potential fault types (FT). Finally, these fault types are then labeled according to the degree of damage. The resulting labeled data is then fed into a supervised system that automatically predicts new data, damaging, or non-damaging.

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

Semi-supervised learning is achieved by incorporating only a limited part of labeled healthy data during training.

For our use case, the model learns the equipment's normal behavior when trained with healthy records. In this way, anomalies can be detected when tested with uncertain data. Hence, if a sample is anomalous, the model should have trouble reconstructing that sample because it does not correspond to the regular behavior which the model learned to reconstruct. A validation set is used to tweak the neural network's hyperparameters. These hyperparameters contain regularization techniques such as dropout and L2 regularization. Obviously, the number of layers and the size of each layer are dependent on one's use case and dataset.

Lastly, the threshold of when a sample is determined as anomalous is automatically defined based on the z-score. This z-score is calculated based on the distribution of healthy data points of a machine (and thus use case dependent). Here, the threshold was chosen based on earlier work (Bonte et al. 2020). By assuming that our healthy data are normally distributed, a value (3σ rule) is chosen as the threshold for each machine. Every sample above the threshold is then considered anomalous, and every record below normal.

In Figure 2, the third block shows the detection of anomalies. The threshold value to do so is automatically determined based on the sigma rule of thumb. Setting a lower threshold facilitates the detection of all anomalies but might also flag too many healthy events as anomalous (false positives). So adjusting the threshold allows for distinguishing between the severity of detected anomalies. Ideally, the value of sigma is optimized to produce the most desirable outcome, i.e. capturing the largest, most obvious anomalies. Furthermore, it is important to analyze and annotate the most anomalous values first and then delve deeper into a space where false positive anomalies are more probable (top–down approach).

A single anomalous sample value is defined as the average of the reconstruction error of each feature. The reconstruction error of a feature is the difference between the value in its input space and the value in its reconstructed space. Healthy samples will be well reconstructed and have a small reconstruction error, while an anomaly will be reconstructed poorly resulting in a bigger reconstruction error. An anomaly's reconstruction error can be dissected into the reconstruction errors of its features and these can then be sorted according to largest impact. This ranking will be different for each sample but trends can emerge when analyzing anomalies this way. Furthermore, anomalies can also be caused by one specific feature reconstruction error that massively outweighs other feature reconstruction errors. Anomalies will be clustered based on this ranking in the next section.

Anomalies clustering

Since many anomalies will be flagged by the auto-encoder, it is convenient to group these into clusters of anomalies with similar characteristics. This clustering technique also allows for accessible labeling, which is discussed in *Section Supervised system*. As the flagged anomalies are high-dimensional, they are in essence complicated to cluster. To solve this issue, the feature space of the anomalies can be transformed from a high dimension to a lower one with techniques such as PCA, UMAP (McInnes, Healy, and Melville 2018), or t-SNE (Van der Maaten and Hinton 2008). The dimensionality reduction approach used in this research is Uniform Manifold Approximation and Projection (UMAP; McInnes, Healy, and Melville 2018), which also lends itself to visualization in three-dimensional space. UMAP has a couple of hyperparameters to tweak, with *n_components* and *n_neighbors* being the most prominent ones. The number of components (*n_components*) directly refers to the output dimension, which is set to three here, allowing us to visually check if the clustering is satisfactory, which is more challenging to audit in a higher-dimensional space.

The number of neighbors (*n_neighbors*) is chosen to be as low as possible, typically being three or four, as a low value for this parameter creates more clusters. The fewer the number of neighbors, the less influence from neighboring samples will be taken into account resulting in more clusters which might have the same source of failure. A large amount of clusters does not produce any issues here because the labeling process is streamlined and thus less time-consuming. Later in the supervised system, samples with the same source of failure but allocated in different clusters will be classified under this same source of failure. If, however, the number of neighbors had been chosen to be high, it would have resulted in big clumps of samples with no clear failure fingerprint, which is unwanted behavior as labeling should be as distinct as possible. Note that the *n_neigbors* hyperparameter is not chosen extremely low, such as one or two, because that would require the expert to label each anomaly individually, which is the reason why clustering is performed in the first place. Hence, an adequate balance is needed when selecting this hyperparameter, and it can be optimized differently for each machine.

Next, this lower-dimensional data (here three dimensions) is clustered using k-means clustering. Note that the presented methodology is not limited to k-means and that for other use cases other clustering approaches, such as Gaussian mixture modeling (GMM) or DBSCAN (Schubert et al. 2017), could be more suitable. However, for our specific use case, we have compared different clustering approaches, such as k-means and GMM, and empirically validated that both algorithms generally produced similar clusters, but k-means yielded more cohesive clusters. Finally, statistics such as the mean and standard deviation are calculated from all the samples within a cluster to evaluate how coherent each cluster individually is. A low standard deviation for a cluster signifies that the clustering is successful for this cluster and the samples within it can be labeled uniformly. Hence, this is an important metric to take into account when labeling these clusters in the next part of our architecture (see *Section Supervised system*).

The feature reconstruction errors calculated in *Section Auto-encoder models* are used to cluster the anomalies. Anomalies with similar poorly reconstructed feature errors probably stem from the same failure source. Hence, we hypothesize that clustering based on these errors will yield satisfactory results and will distinguish failure fingerprints from one another.

The k-means clustering technique automatically chooses the best k, i.e. the number of clusters. This best k is determined by incrementing the number of clusters and evaluating the inertia (i.e. sum of squared distances) of each number of clusters. During initialization, the initial cluster centroids are selected using sampling based on an empirical probability distribution of the points' contribution to the overall inertia (k-means++). Next, the implementation makes several trials at each sampling step and chooses the best centroid among them (greedy k-means++).

However, this process is not seamless as it can still benefit from manual verification. If the clustering algorithm has wrongfully joined clusters, the process always has the manual check to pass through and correct this error. The manual verification is simplified by having the three-dimensional UMAP plot to visually distinguish the clusters. This manual verification step is implemented as a fallback step so that domain experts can check how the algorithm clustered the data. Instead of visual inspection, techniques such as

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

silhouette-coefficient could also be used to validate the clustering procedure.

An exemplary result of dryer D001 is shown in Figure 3. The different colors signify the distinct cluster groups after applying k-means on the UMAP dimensional reduced data. The large crosses within the different cluster groups show the cluster mean of that group. One can notice clear-cut clusters with cluster means in the center of the group for this particular case. As each machine produces a different subset of anomalies, this could lead to more vague clusters for other machines. This phenomenon is present in Figure 4, where the right-side clusters have lower intra-cluster distances, while the left-side clusters have higher intra-cluster distances. These latter clusters will have a high standard deviation for their feature values and will thus be harder to label.

Supervised system

As pointed out earlier, turning an unsupervised system (*Section Auto-encoder models*) into a supervised system utilizing AE advantages results in a more resilient architecture with not much adjustment needed. The main benefit of this transformation is the convenience of evaluating the models, which cannot be done in an unsupervised learning context since no ground truth is available. Our research attempts to solve interpretability issues by leveraging the newly labeled qualitative data with these supervised models. Based on the principle of Occam's razor, these simple models are leveraged to gain more insights into the core of the faults. Ultimately, we hope to advocate the usage of ML techniques in the manufacturing industry in this way.

The results gained from supervised learning are far more reliable when clarifying an anomaly and linking it to a mechanical failure. However, this requires the anomalies to be labeled, demanding a considerable amount of cumbersome manual labor. As a result of the clustering techniques in Section Anomalies clustering, this process has been simplified by examining the clusters' statistics. When the standard deviation of a cluster is minuscule, then the mean of that group can be seen as a direct interpretation of all samples within that cluster. Consequently, labeling can be

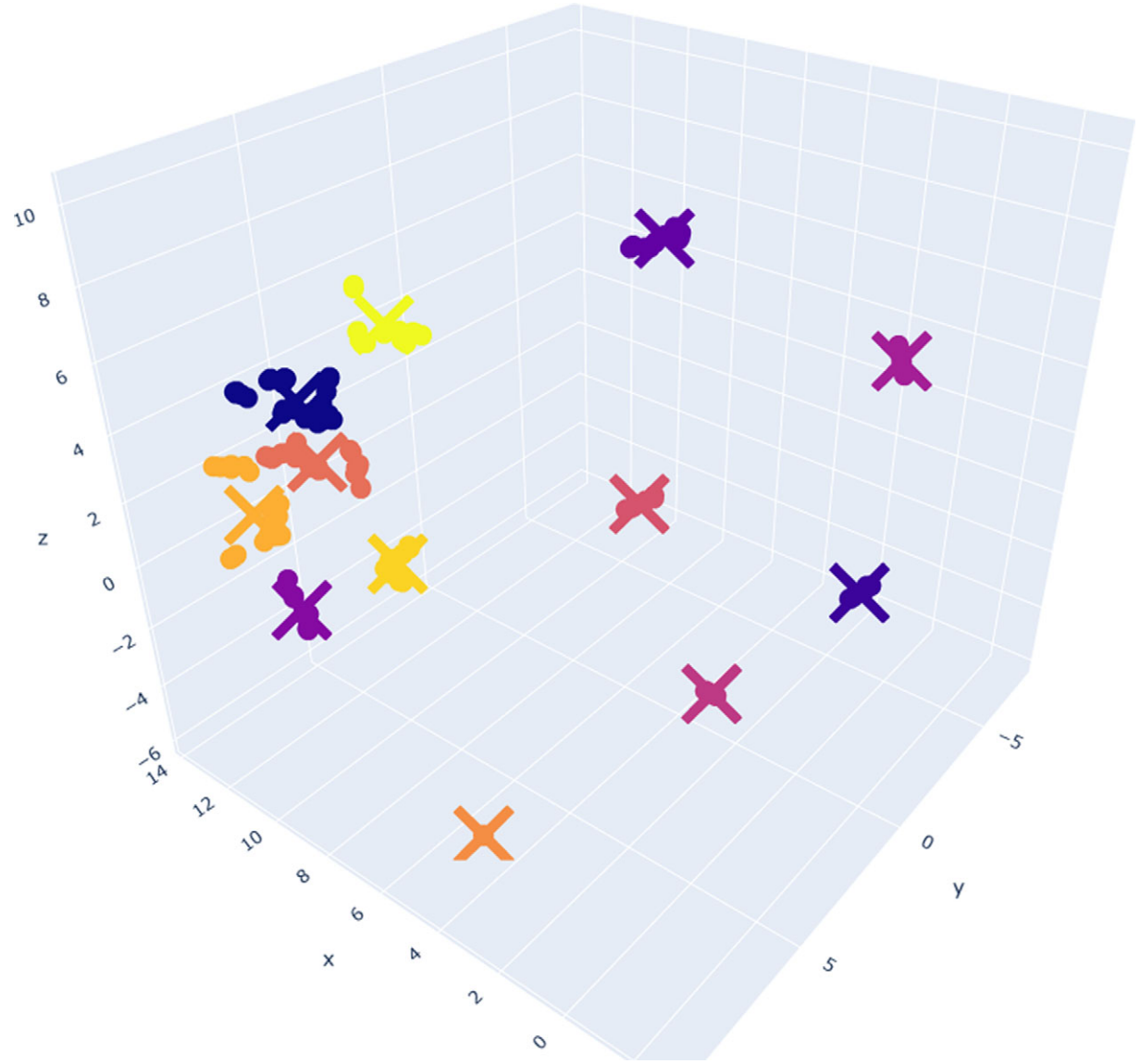

Figure 3. Clusters of anomalies from dryer D001. This machine has distinct clusters which will facilitate labeling.

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

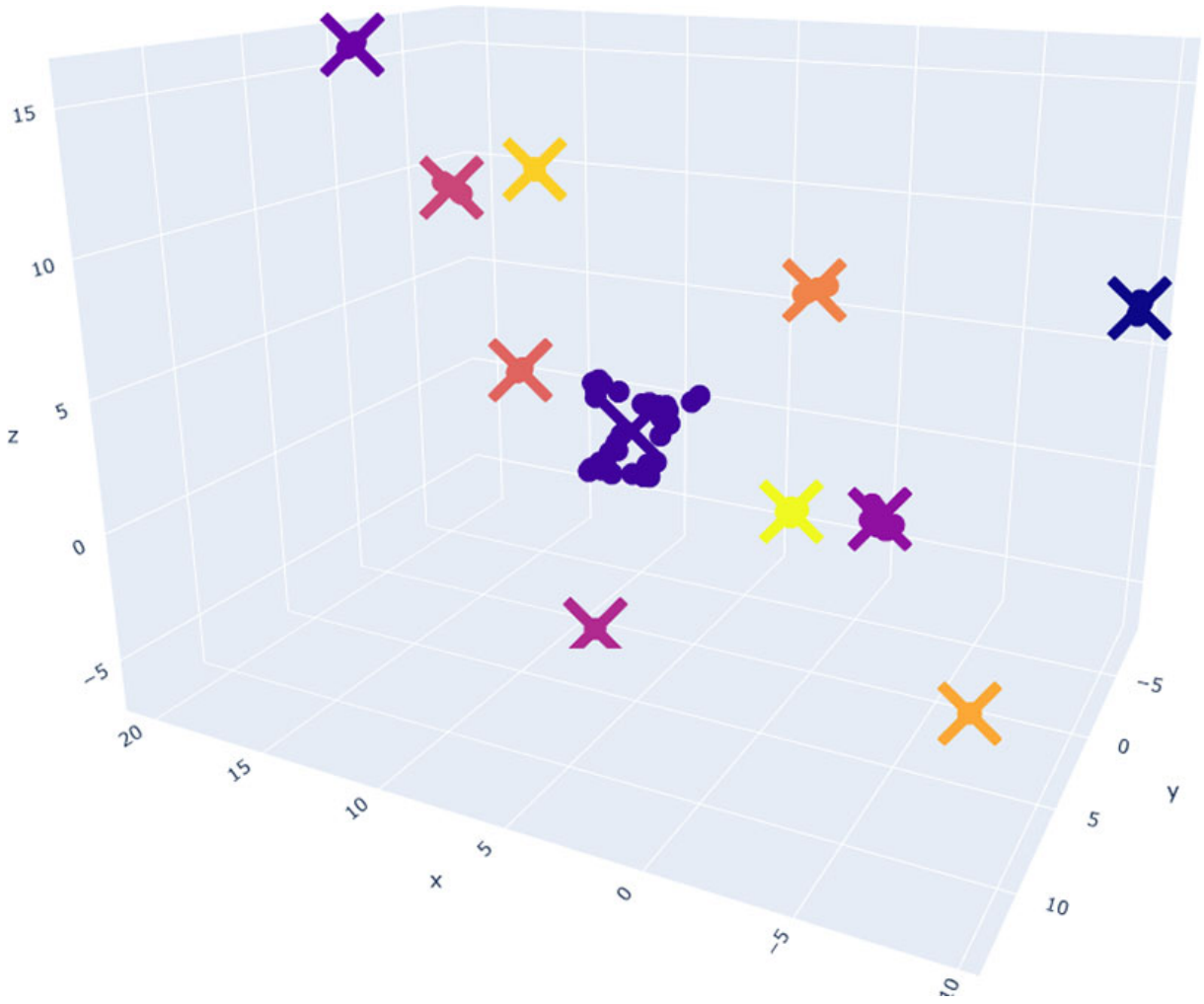

Figure 4. Clusters of anomalies from filter F017. This machine has more vague clusters than Figure 3 which will hinder labeling.

performed based on only the group's mean, reducing the overhead of labeling the anomalous samples individually. Thus, our research makes as few assumptions as possible about the data and labels the data through domain experts *after* clustering via a user-friendly dashboard shown in Figure 5 which was made with Plotly Dash. Using this dashboard, these experts can easily select the cluster and events within that cluster to label an anomaly along with a description of that anomaly. Several details such as the mean and standard deviation of these clusters can be investigated. Additionally, the raw signals along with the highlighted anomalous windows are also displayed in an orderly manner, only showing the relevant signals for that cluster.

Here, we focused on obtaining two kinds of labels either damaging or non-damaging, and the specific fault type. The first kind of label distinguishes an anomaly as either damaging or non-damaging. This binary classification acts as a first rudimentary barrier to disregard all incorrectly flagged anomalies by the auto-encoder, the so-called false positives. The provided domain expert knowledge should ascertain that the anomalies are not damaging and thus not interesting to be detected. Moreover, when an anomaly is regarded as nondamaging, feedback can be given to the auto-encoder that has falsely flagged this anomaly. The auto-encoder can then be optimized to learn to reconstruct these samples so they are classified as healthy when it is run again. In case an anomaly is indeed damaging, a further distinction can be made. Therefore, the second kind of label differentiates each anomaly from its origin of fault (fault type). For example, a type of fault could be a high-pressure surge in a specific chamber of equipment or an operator fault.

Using the labeled data, the supervised models are trained to learn the behavior that experts denote and assign the most suitable label to each anomaly. By ascribing a non-damaging label to a sample, all false positives of the auto-encoder should eventually be detected. Also, more labeled samples lead to better performance, ideally resulting in an autonomous architecture. Autonomy of the architecture is achieved when all fault types are detected and the auto-encoder can perfectly distinguish the difference between them.

By now, a lot of supervised models exist that perform well in fault classification (Kotsiantis, Zaharakis, and Pintelas 2006). The purpose of this research is not to improve on the SOTA of (and differences between) these models. The models are merely used here to see if anomaly fingerprints are classifiable. Hence, without loss of generality, we employ simple logistic regression models in our research to demonstrate the pipeline's functionality. However, this classifier can easily be replaced with more complex models,

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

Figure 5. Example of the labeling dashboard with an anomaly of filter F017. This cluster of anomalies is caused by a current sensor having odd variance and standard deviation values within the window(s) shown in orange. Hence, the cluster is labeled by an expert as damaging behavior along with a small description.

such as random forests, SVMs, or neural networks, depending on the specific use case and/or dataset characteristics.

The labeled anomalies are split into a test and train set before training the logistic regression model. These anomalies are split in a stratified fashion, taking the class labels (damaging and non-damaging) into account, resulting in balanced splits. The test set size contains one-third of the total labeled anomalies, while training is performed on the remaining two-thirds. Hyperparameter optimization is not required in our use case since we already generate satisfactory results with a fixed hyperparameter subset. Also, even though our data is sequential in nature, due to the statistical descriptors we use to capture sliding window information, we do not consider the non-sequential train-test split to result in temporal data leakage in our application.

Table 2 describes the evaluation of a test set of these labeled anomalies for each examined machine. Several metrics such as weighted accuracy, precision, recall, and F1 score are shown. These metrics are only reported for the positive class, ergo the damaging anomalies. Finally, the support, which means the number of labeled anomalies, is given for the damaging, non-damaging, and total of both classes.

Even with a limited number of training samples, the models already perform adequately. Especially the negative class, the non-damaging class, performs nearly perfectly due to class imbalance

Table 2. Evaluation of the logistic regression model of all machines

					Support[a]		
Machine	Weighted accuracy	Precision	Recall	F1 score	Non-damaging	Damaging	Total
D001	1.00	1.00	1.00	1.00	26	3	29
D009	0.91	0.91	0.91	0.91	12	11	23
D020	0.90	1.00	0.80	0.89	4	5	9
F008	1.00	1.00	1.00	1.00	8	4	12
F012	1.00	1.00	1.00	1.00	17	1	18
F017	0.72	1.00	0.44	0.62	44	9	53

[a]Support indicates the amount of anomalies that are flagged by the auto-encoder that constitutes the test set.

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

favoring this (negative) class. Conversely, the positive class, the damaging class, performs worse due to the relative lack of samples within that class. Therefore, inspecting the precision and recall of the positive class is insightful here.

General results

Whereas so far only results of the different components were presented, here, some general results of the entire architecture are mentioned. The positive opportunities gained from this research offer a direct solution to the problems discussed in current related work: preprocessing problems, use of unsupervised data and measuring accuracy, explainability, and labeling overhead.

First, a closer look is taken at the anomalies and their respective fault type. As mentioned before, the AEs produce many anomalies. Therefore, a sunburst plot is generated for each machine that shows which portion of the anomalies are damaging and which are not. Examples of such a sunburst plot are depicted in Figures 6 and 7. The center circle shows the machine along with the total amount of anomalies found by the AE for that machine. The middle ring elaborates on the health of the anomalies, being damaging or non-damaging. The outer ring focuses on the window and stride sizes on which the auto-encoder was trained and helps to characterize the anomalies, such as examining which intervals lead to more anomalous behavior.

The damaging anomalies are further divided into specific fault types if possible. These fault types were identified and verified by operators and/or experts of the machines. Examples of these fault types are high-pressure surges, temperature spikes, high oscillating vibrations, large current draws, and other damaging behavior. This means that earlier unnoticed faults of all sorts of origins can now be detected using our pipeline, and the architecture is not restricted to one specific fault type but is able to learn to detect all different fault labels as given by the operators.

This way, with little labeling resources, the system has transformed from unsupervised learning to a semi-supervised architecture with satisfactory results as confirmed by the end users.

Also, the explainability of the architecture should be examined. The transformation from an unsupervised system to a semi-supervised system was done by incorporating logistic regression models. One significant advantage of logistic regression compared to other classifier techniques is the interpretability of the coefficients. The coefficients give a measure of the most prominent features used to distinguish fault types from normal behavior.

An example is filter F008 where a particular fault causes the temperature of the machine to increase and decrease rapidly, causing damaging impact to that machine or system. When looking at the first six logistic regression coefficients of that fault type (Figure 8), four of them are related to the temperature sensor within that machine. Particularly, the fluctuation of values from that sensor is alarming as the two highest-ranked features are based on variances.

Thanks to our architecture, the resources needed to perform the labeling tasks were heavily reduced. By using a user-friendly

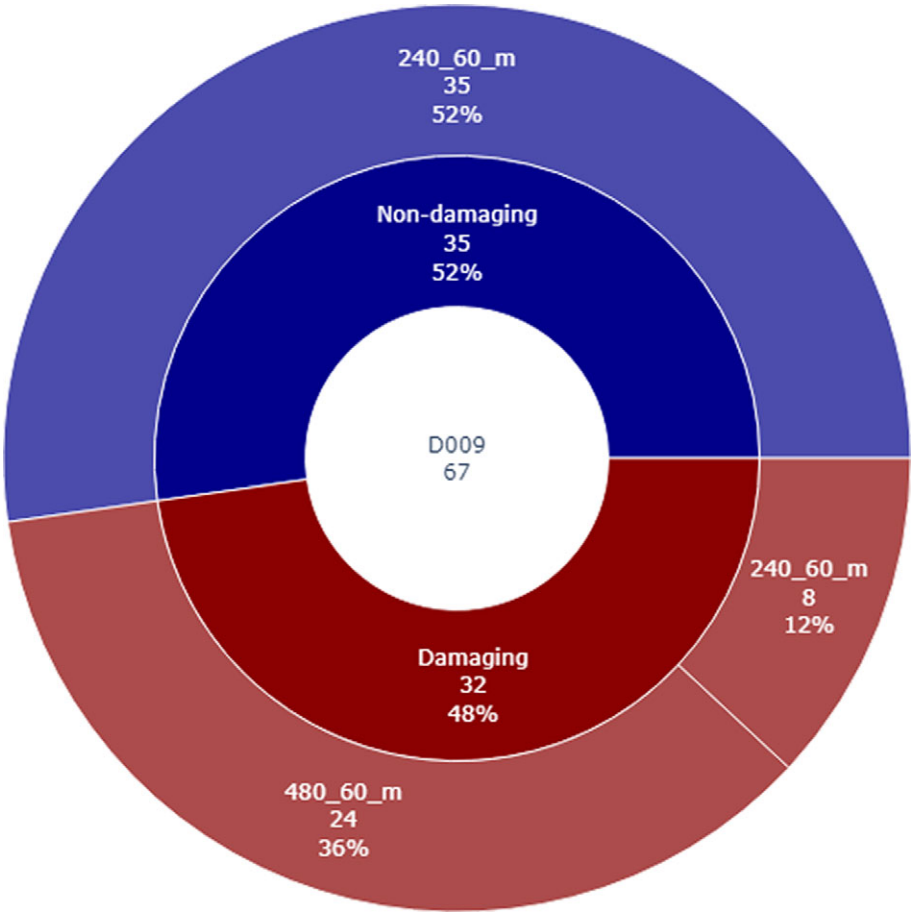

Figure 6. Sunburst plot of dryer D009.

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

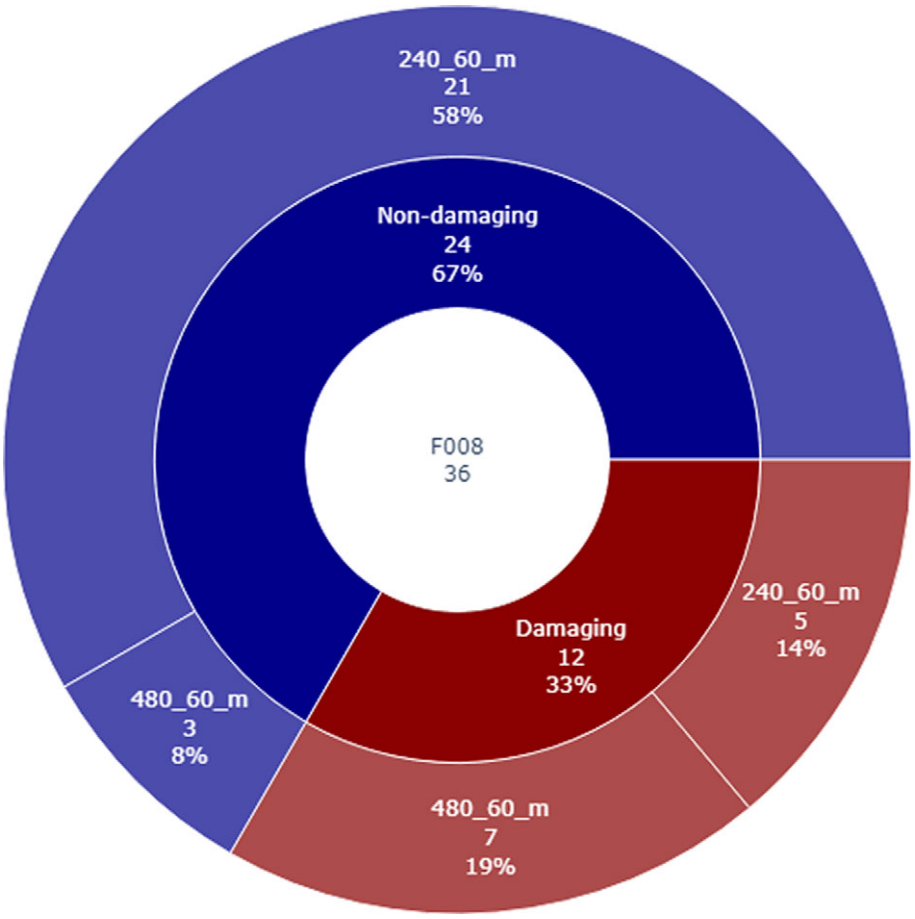

Figure 7. Sunburst plot of filter F008.

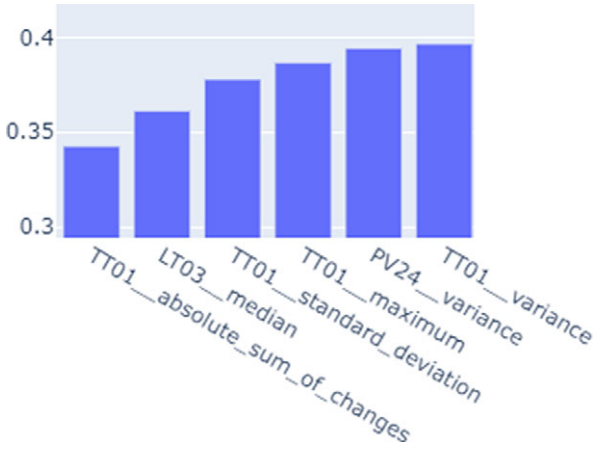

Figure 8 The most impactful logistic regression coefficients of a temperature fault of filter F008. Four of the six coefficients are related to the fluctuation of a temperature sensor.

dashboard (Figure 5), an expert could efficiently input his knowledge into the architecture. Also, the logistic regression coefficients can be investigated in the dashboard to ideally deduce the fault types and their origins.

Furthermore, Table 3 gives more details on the amount of samples, anomalies, clusters, and the reduction of labeling resources. The number of samples per machine equals the total amount of rolling window samples after preprocessing before training the auto-encoder model. A part of these samples are then flagged by the auto-encoder model as anomalies while domain experts divide them further in the damaging or non-damaging classes. This labeling task is facilitated by labeling the clusters that are generated by clustering the total amount of anomalies per machine based on similar fingerprints.

The right part of Table 3 reports on the reduction of labeling resources and also mentions the labeling gain when compared to other labeling tasks. Finally, the last column shows us this main labeling gain by dividing the number of clusters to label, with the number of anomalies found by the auto-encoder, and then subtracting this result from one, also seen in Formula 1. This formula represents the advantages gained that otherwise needed to be labeled in other research. The *w/s* notation refers to the specific window and stride sizes all the anomalies were extracted. The only data that needs to be labeled is at best only the clusters or worst case all the anomalies within each cluster, the latter would result in no labeling gain. The labeling gain of all machines is at best around 80% to 90%.

$$1 - \frac{\#\text{Clusters}}{\sum_{w/s=0}^{n} \#\text{Anomalies}_{w/s}} \tag{1}$$

Finally, we want to emphasize the adaptability of the pipeline and clarify that the components mentioned are illustrative rather than fixed choices. The data processing steps, feature subset

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press

Table 3. General description of the number of samples, anomalies, and clusters of all machines. The percentages given in the last three columns give more insight into the amount of data needed to label when compared to labeling the total dataset

		# Anomalies						
Machine	# Samples	Non-damaging	Damaging	Total	# Clusters	$\frac{\#Anomalies}{\#Samples}$	$\frac{\#Clusters}{\#Samples}$	Labeling gain[a]
D001	15,250	76	10	86	10	0.56%	0.066%	88.37%
D009	18,713	35	32	67	11	0.36%	0.059%	83.58%
D020	5,534	11	15	26	5	0.47%	0.090%	80.77%
F005[b]	28,827	95	2	97	12	0.34%	0.042%	87.63%
F008	26,319	24	12	36	8	0.14%	0.030%	77.78%
F012	24,672	48	4	52	9	0.21%	0.036%	82.69%
F017	26,164	132	28	160	12	0.61%	0.046%	92.50%

[a]The labeling gain of a machine is equal to the number of clusters divided by the number of anomalies found by the auto-encoder and then subtracted from one, also represented in Formula 1.
[b]Note that a lack of damaging anomalies prohibits the usage of ML techniques on filter F005. Only two anomalies were labeled as damaging so a proper division in train and set cannot be performed.

selection, dimensionality reduction technique, and clustering method should all be tailored to the specific needs of the reader's use case.

Conclusions

The breakdown of machinery is a costly expense for many manufacturing industries. Insufficient labels limiting them to unsupervised AD techniques often hold companies back from solving this issue with predictive maintenance. The lack of interpretability of ML models also leads to unjust reluctance to use these models. Therefore, this research proposed a threefold architecture to tackle these limitations. The designed auto-encoder models are the backbone that detects anomalous behavior. Clustering these anomalies then limits the overhead of labeling each anomaly individually. After labeling the clusters of similar fingerprints, a model is learned that mimics experts' knowledge and identifies faults that show behavior similar to that that has been seen before. Eventually, this architecture leads to a higher certainty of identifying the correct fault type. This is demonstrated with a real-life use case in association with a forerunner of the pharmaceutical industry, indicating the need for such valuable research. The results of this research look promising as the architecture is indeed able to recognize critical recurring faults with little labeling overhead, having a labeling gain of 90% at best in some cases using our methodology.

Concerning future work, we will look into improving the feedback loop to the auto-encoder models to boost the AE performance when fed with expert knowledge.

Data availability. Data used within this research project contains sensitive information and can thus not be shared.

Author contribution. All authors contributed to the study's conception and design. Software development, data manipulation, and analysis were performed by Colin Soete. The first draft of the manuscript was written by Colin Soete and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding statement. No grants were awarded for this research.

Competing interest. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Code availability. The code written for this research project can be transformed into an open-source library if desired.

Selected feature subset list. Listing 1. The feature subset selected specifically for our use case using the tsfresh library. The description along with the specific implementation of each feature can be found on the documentation page of the library: https://tsfresh.readthedocs.io/en/latest/text/list_of_features.html

```
feature_subset = {
    " absolute_sum_of_changes":
    None,
    "maximum": None,
    "mean": None,
    "mean_change": None,
    "median": None,
    "minimum": None,
    "percentage_of_reoccurring
    _values_to_all_values": None,
    " ratio_beyond_r_sigma":
    [{"r": 3}],
    " ratio_value_number_to_time
    _series_length": None,
    "skewness": None,
    " standard_deviation ": None,
    "variance " : None,
    }
```

References

Bonte Pieter, **Sander Vanden Hautte**, **Annelies Lejon**, **Veerle Ledoux**, **Filip De Turck**, **Sofie Van Hoecke, and Femke Ongenae**. 2020. Unsupervised anomaly detection for communication networks: an autoencoder approach. In *Iot streams for data-driven predictive maintenance and iot, edge, and mobile for embedded machine learning*, edited by Joao Gama, **1325**: 160–172. Ghent, Belgium: Springer. ISBN: 9783030667696. https://doi.org/10.1007/978-3-030-66770-2_12.

Chakraborty Supriyo, **Richard Tomsett**, **Ramya Raghavendra**, **Daniel Harborne**, **Moustafa Alzantot**, **Federico Cerutti**, **Mani Srivastava**, et al. 2017. Interpretability of deep learning models: a survey of results. In *2017 IEEE Smartworld, ubiquitous intelligence computing, advanced trusted computed, scalable computing communications, cloud big data computing, internet of people and smart city innovation (smartworld/scalcom/uic/atc/cbdcom/iop/sci)*, 1–6. https://doi.org/10.1109/UIC-ATC.2017.8397411.

Christ Maximilian, **Nils Braun**, **Julius Neuffer, and Andreas W. Kempa-Liehr**. 2018. Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package). *Neurocomputing* **307**:72–77. ISSN: 0925-2312. https://doi.org/10.1016/j.neucom.2018.03.067. https://www.sciencedirect.com/science/article/pii/S0925231218304843.

De Paepe Dieter, **Sander Vanden Hautte**, **Bram Steenwinckel**, **Filip De Turck**, **Femke Ongenae**, **Olivier Janssens, and Sofie Van Hoecke**. 2020. A generalized matrix profile framework with support for contextual series analysis. *Engineering Applications of Artificial Intelligence* **90**:103487. ISSN: 0952-1976. https://doi.org/10.1016/j.engappai.2020.103487. https://www.sciencedirect.com/science/article/pii/S0952197620300087.

Fährmann Daniel, **Naser Damer**, **Florian Kirchbuchner, and Arjan Kuijper**. 2022. Lightweight long short-term memory variational auto-encoder for multivariate time series anomaly detection in industrial control systems. *Sensors* **22** (8): 2886.

Gittler Thomas, **Magnus Glasder**, **Elif Öztürk**, **Michel Lüthi**, **Lukas Weiss, and Konrad Wegener**. 2021. International conference on advanced and competitive manufacturing technologies milling tool wear prediction using unsupervised machine learning. *The International Journal of Advanced Manufacturing Technology* **117**, no. 7 (December): 2213–2226. ISSN: 1433-3015. https://doi.org/10.1007/s00170-021-07281-2.

Hindy Hanan, **Robert Atkinson**, **Christos Tachtatzis**, **Jean-Noël Colin**, **Ethan Bayne, and Xavier Bellekens**. 2020. Towards an effective zero-day attack detection using outlier-based deep learning techniques. https://doi.org/10.48550/arXiv.2006.15344.

Kotsiantis Sotiris B, **Ioannis D Zaharakis, and Panayiotis E Pintelas**. 2006. Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review* **26** (3): 159–190.

Li Jinbo, **Hesam Izakian**, **Witold Pedrycz, and Iqbal Jamal**. 2021. Clustering-based anomaly detection in multivariate time series data. *Applied Soft Computing* **100**:106919. ISSN: 1568-4946. https://doi.org/10.1016/j.asoc.2020.106919. https://www.sciencedirect.com/science/article/pii/S1568494620308577.

Li Xiang, **Xu Li, and Hui Ma**. 2020. Deep representation clustering-based fault diagnosis method with unsupervised data applied to rotating machinery. *Mechanical Systems and Signal Processing* **143**:106825. ISSN: 0888-3270. https://doi.org/10.1016/j.ymssp.2020.106825. https://www.sciencedirect.com/science/article/pii/S0888327020302119.

Li Xuhong, **Haoyi Xiong**, **Xingjian Li**, **Xuanyu Wu**, **Xiao Zhang**, **Ji Liu**, **Jiang Bian, and Dejing Dou**. 2022. Interpretable deep learning: interpretation, interpretability, trustworthiness, and beyond. *Knowledge and Information Systems* **64** (12): 3197–3234.

Li Zhe, **Jingyue Li**, **Yi Wang, and Kesheng Wang**. 2019. A deep learning approach for anomaly detection based on sae and lstm in mechanical equipment. *International Journal of Advanced Manufacturing Technology* **103**, no. 1 (July): 499–510. ISSN: 1433-3015. https://doi.org/10.1007/s00170-019-03557-w.

Lu Yue, **Thirumalai Vinjamoor Akhil Srinivas**, **Takaaki Nakamura**, **Makoto Imamura, and Eamonn Keogh**. 2023. Matrix profile xxx: madrid: a hyper-anytime and parameter-free algorithm to find time series anomalies of all lengths. In *2023 ieee international conference on data mining (icdm)*, 1199–1204. IEEE. https://doi.org/10.1109/ICDM58522.2023.00148.

McInnes Leland, **John Healy, and James Melville**. 2018. Umap: uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Mulders Michel, and Mark Haarman. 2017. *Predictive maintenance 4.0: predict the unpredictable.* Edited by PwC. Accessed March 6, 2021. https://www.pwc.be/en/documents/20171016-predictive-maintenance-4-0.pdf.

Nasir Vahid, and Farrokh Sassani. 2021. A review on deep learning in machining and tool monitoring: methods, opportunities, and challenges. *The International Journal of Advanced Manufacturing Technology* **115**, no. 9 (August): 2683–2709. ISSN: 1433-3015. https://doi.org/10.1007/s00170-021-07325-7.

Ochella Sunday, **Mahmood Shafiee, and Chris Sansom**. 2021. Adopting machine learning and condition monitoring pf curves in determining and prioritizing high-value assets for life extension. *Expert Systems with Applications* **176**:114897. ISSN: 0957-4174. https://doi.org/10.1016/j.eswa.2021.114897. https://www.sciencedirect.com/science/article/pii/S0957417421003389.

Pang Guansong, **Chunhua Shen**, **Longbing Cao, and Anton Van Den Hengel**. 2021. Deep learning for anomaly detection: a review. *ACM computing surveys (CSUR)* **54** (2): 1–38.

Pu Guo, **Lijuan Wang**, **Jun Shen, and Fang Dong**. 2020. A hybrid unsupervised clustering-based anomaly detection method. *Tsinghua Science and Technology* **26** (2): 146–153.

Schubert Erich, **Jörg Sander**, **Martin Ester**, **Hans Peter Kriegel, and Xiaowei Xu**. 2017. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)* **42** (3): 1–21.

Serradilla Oscar, **Ekhi Zugasti**, **Julian Ramirez de Okariz**, **Jon Rodriguez, and Urko Zurutuza**. 2021. Adaptable and explainable predictive maintenance: semi-supervised deep learning for anomaly detection and diagnosis in press machine data. *Applied Sciences* **11** (16). ISSN: 2076-3417. https://doi.org/10.3390/app11167376. https://www.mdpi.com/2076-3417/11/16/7376.

Van der Maaten Laurens, and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research* **9** (86): 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html.

Weichert Dorina, **Patrick Link**, **Anke Stoll**, **Stefan Rüping**, **Steffen Ihlenfeldt, and Stefan Wrobel**. 2019. A review of machine learning for the optimization of production processes. *The International Journal of Advanced Manufacturing Technology* **104** (5): 1889–1902.

Yan Jihong, **Yue Meng**, **Lei Lu, and Lin Li**. 2017. Industrial big data in an industry 4.0 environment: challenges, schemes, and applications for predictive maintenance. *IEEE Access* **5**:23484–23491. https://doi.org/10.1109/ACCESS.2017.2765544.

https://doi.org/10.1017/S0890060424000131 Published online by Cambridge University Press