

Unsupervised Anomaly Detection in Predictive Maintenance using Sound Data*

Antonino Ferraro¹, Antonio Galli¹, Valerio La Gatta¹, Vincenzo Moscato¹, Marco Postiglione¹, Giancarlo Sperli¹ and Francesco Moscato²

¹Department of Electrical Engineering and Information Technology, University of Naples Federico II, Italy

²Department of Information Engineering, Electrical Engineering and Applied Mathematics, University of Salerno, Italy

Abstract

This paper represents an extended abstract of a recent proposal ([1]), in which the authors present a new methodology for unsupervised anomaly detection in predictive maintenance using sound data. In particular, the methodology leverages LSTM and CNN-based autoencoders to process continuous audio streams from different audio sources in real-world factories based on a customized sliding window. The novelties of the proposed approach include a general methodology for unsupervised anomaly detection, machine ID encoding using one-hot encoding, and conditioning an autoencoder by jointly analyzing the relationships between the mel-spectrogram and the machine ID to compute an anomaly score. The methodology achieves good performances in terms of effectiveness and in addition low inference time and memory requirements.

Keywords

Deep Learning, Anomaly Detection, Industrial AI, Predictive maintenance

1. Introduction

Over the past two decades, *Anomalous Sound Detection* (ASD) has become an increasingly challenging task in various applications. ASD is used to identify whether the sound emitted from an object is normal or anomalous, and early detection can prevent critical problems ([2, 3]). Traditional approaches use supervised machine learning models ([2, 4]), while unsupervised models have also been used to distinguish between normal and abnormal situations ([5, 6]). Recently, deep learning approaches have been successfully exploited in various contexts ([7, 8]). In industrial applications, existing anomaly detection systems rely on monitoring sensors, with most systems utilizing visual detection methods. However, these methods have limitations such as illumination or occlusion, which can hinder their performance, particularly in real-time applications. Additionally, sensor-based analysis systems can be vulnerable to deception, as

SEBD 2023: 31st Symposium on Advanced Database System, July 02–05, 2023, Galzignano Terme, Padua, Italy

* You can use this document as the template for preparing your publication. We recommend using the latest version of the ceurart style.

✉ antonino.ferraro@unina.it (A. Ferraro); antonio.galli@unina.it (A. Galli); valerio.lagatta@unina.it (V.L. Gatta); vincenzo.moscato@unina.it (V. Moscato); marco.postiglione@unina.it (M. Postiglione); giancarlo.sperli@unina.it (G. Sperli); fmoscato@unisa.it (F. Moscato)

🆔 0000-0002-1326-0325 (A. Ferraro); 0000-0001-9911-1517 (A. Galli); 0000-0002-0877-7063 (V.L. Gatta); 0000-0002-0877-7063 (V. Moscato); 0000-0002-0877-7063 (M. Postiglione); 0000-0003-4033-3777 (G. Sperli); 0000-0002-7056-4579 (F. Moscato)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

demonstrated by the Triton malware and Stuxnet virus. As a result, there is growing interest in the use of additional features, such as audio streams, that can be analyzed externally by the system to improve performance and reliability.

Various Anomalous Sound Detection (ASD) systems have been developed to address the limitations of traditional techniques ([2, 9, 10]). These systems use acoustic data features, such as Mel-scale¹ spectrograms or air pressure values, to train neural networks, typically using autoencoder architectures ([11]). Studies have compared the performance of different autoencoder architectures for ASD, including Convolutional LSTM, sequential Convolutional Autoencoder, LSTM-based autoencoder, dense and convolutional architectures, Transformer-based and Conformer-based autoencoder ([12]). A few-shot approach, called SNIPER, has been designed by ([13]) to overcome the problem of insufficient observed anomalies. However, additional information related to the equipment, such as equipment ID, can potentially improve the patterns learned from the model.

This paper represents an extended abstract of a recent proposal [1], in which the authors present a new methodology for unsupervised anomaly detection in predictive maintenance using sound data. The methodology is flexible and efficient for real-world scenarios, allowing it to be applied to multiple instances of the same or different equipment, and can instantiate different types of autoencoders. The authors evaluate the methodology using LSTM and CNN-based autoencoders to process continuous audio streams from different audio sources in real-world factories based on a customized sliding window. The novelties of the proposed approach include a general methodology for unsupervised anomaly detection, machine ID encoding using one-hot encoding, and conditioning an autoencoder by jointly analyzing the relationships between the mel-spectrogram and the machine ID to compute an anomaly score. The methodology achieves low inference time and memory requirements and has been evaluated on audio streams from multiple machine types. Finally, the proposed methodology enables application in real scenario by analyzing continuous audio stream coming from different audio sources on the basis of a customized sliding windows.

2. Methodology

The Anomaly Detection task is a challenging aspect of predictive maintenance, which involves identifying anomalous warning or failure states in industrial machines to improve maintenance scheduling. Various sensor-based techniques have been proposed for addressing this task. However, cyber-physical attacks like Triton or Stuxnet have raised new challenges that require the investigation of novel features, including equipment sounds. This article proposes a methodology for addressing the Anomaly Detection task, consisting of an Offline Training and Online Operation phase. These phases are discussed in Sections 2.1.1 and 2.1.2.

¹The Mel-Scale is a frequency scale that segments the entire frequency spectrum into x uniformly spaced frequency bins. The term 'evenly spaced' is used to denote that the separation of frequency bins along the frequency dimension more closely approximates the sensitivity of the human auditory system compared to the linearly spaced frequency bands that are typically employed in spectrograms.

2.1. Methodology Description

The proposed *ASD* methodology is composed by two main phases: an offline phase (Figure 1), aiming to train an autoencoder model on the basis of extracted features from a pre-collected normal audio clips (Figure 2), and an online operation phase (Figure 3), that supports analysis and detection in real scenario.

2.1.1. Offline Training Phase

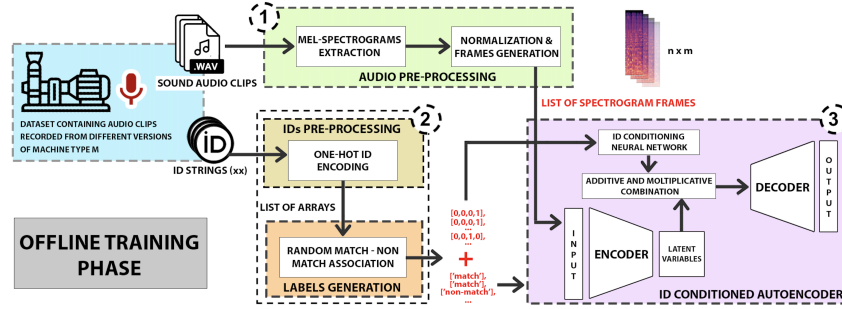


Figure 1: Overview of offline training phase of the proposed ASD system.

The offline training phase includes three modules: Audio Pre-processing, IDs Pre-processing, and ID Conditioned autoencoder. The Audio Pre-processing consists of two components, Mel-Spectrogram extractor and Normalization and Frames generator, and extracts features from audio signals. The IDs Pre-processing encodes the ID string code of each machine version. The ID Conditioned autoencoder jointly analyzes the outputs of the first two modules and computes an anomaly score using an encoder-decoder architecture. Mel-Spectrogram extractor produces log-mel-scale images, and Normalization and Frames generator segments them into overlapping frames.

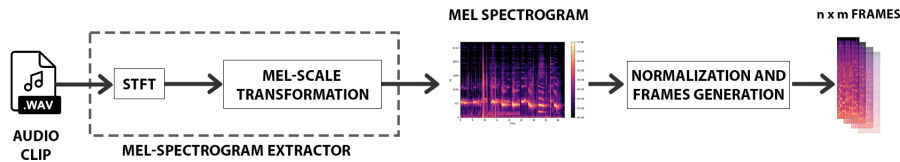


Figure 2: Features extraction block

Fig. 2 outlines the pertinent parameters for feature extraction from an audio signal using Short-Time Fourier Transform (STFT), including the window length (n_{fft}), window overlap (hop_length), and number of Mel scale bins (n) used for spectrogram transformation.

The ID pre-processing step employs One-Hot Encoding to convert machine IDs to binary sequences of equal length, allowing the encoder-decoder architecture to differentiate between sound signals from different machine versions. This ensures that frames of the same spectrogram

are associated with the same binary sequence. The ID binary sequences are incorporated into the training process via the ID Conditioned Autoencoder module.

This module consists of an autoencoder and an ID Conditioning Neural Network. The autoencoder uses an encoder-decoder architecture to reconstruct input spectrogram frames. The encoder encodes the input into a latent representation, which is then reconstructed by the decoder. The ID Conditioning Neural Network maps one-hot-encoded ID arrays into conditioning functions that are combined with the output of the decoder. The goal of ID conditioning is to inform the model about the presence of different machines for the recognition of their different normal behaviors.

We introduce concatenation to reduce the number of false negatives because normal sound of a machine M with ID w could be different from normal sound of a machine M with ID z this could generate some false negatives.

The key concept is that the autoencoder must be trained to reconstruct normal audio spectrograms in input only if the provided ID is correct. With this assumptions, after the training, if a normal test sample is placed in input, a low reconstruction error (in terms of mean absolute error or mean squared error) is expected, while if there is an anomalous one, an high reconstruction error is generated, even if this anomalous behavior is similar to a normal behaviour of another machine. The similarity problem is so resolved by the presence of the ID.

Nevertheless, the training process needs to be revised for supporting the autoencoder in recognizing the relationships between machine identifiers (IDs) and audio signal, because the ID conditioning in latent space is not enough. For this reason, the *Label Generation* module randomly changes with a probability $1 - \alpha$ the correct *ID* binary sequence associated to an audio signal with another one available. In particular, it adds the string *match* or *not-match* (corresponding to the output of the *Random Match - Non Match association* module) for each frame associated to the same audio clip on the basis of decisions.

Furthermore, a new loss must be used and tuned in the training process because the classical difference between the encoder input and the decoder output is not enough because the association between ID and audio signal may not be correct.

2.1.2. Online Operation Phase

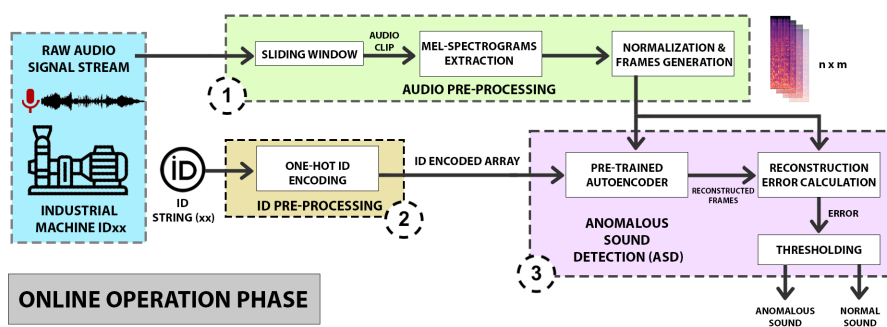


Figure 3: Overview of online operating phase of the proposed ASD system.

During the operational phase, our approach should be capable of processing a continuous audio stream from machines using sliding windows. Figure 3 illustrates the architecture for online operation, where the left side analyzes the raw audio signal stream using sliding windows. The sliding window component samples the last T seconds from the stream every h seconds. The extracted audio signals are processed through the mel-spectrogram extractor and the frame generator, whose details are explained in the offline phase. The goal of the *Anomalous Sound Detection* is to classify a sound signal as normal or anomalous. This subsystem consists of the *Pre-trained Autoencoder*, the *Reconstruction Error Calculator*, and the *Thresholding*.

3. Experimental Evaluation

Our aim is to investigate the sound analysis to deal with the more recent cyber-physical attacks, whose aim is to deceive monitoring platform affecting the performance of classical predictive maintenance tools.

In this section we described the experimental evaluation of the proposed approach in terms of efficacy and efficiency on the *DCASE* dataset, whose characterization is shown in Section 3.1. We further discussed pre-processing phase for generating images from audio signals (see Section 3.2) and autoencoder structure, also optimized the related hyperparameter (see Section 3.4). Finally, performance metrics are described in Section 3.5.

Two types of autoencoders are considered in order to investigate the ID conditioning effects, also analyzing its compatibility with different encoding and decoding processes. According to their autoencoder's models, the overall architectures are identified as *ID Conditioned LSTM Autoencoder (IDC-LSTM-AE)* and *ID Conditioned Convolutional Autoencoder (IDCCAE)*, that are implemented and trained on four machines available in *DCASE* dataset.

3.1. Dataset and Recording Procedure

We evaluated our methodology on the *Unsupervised Detection of Anomalous Sounds for Machine Condition Monitoring* dataset, provided by DCASE 2020 TASK 2 belonging to MIMII dataset ([14]). In particular, it contains audio clips recorded from four different machine types (pumps, valves, slide rails and fans), each one composed by four different versions.

In conclusion, four models have been trained, one for each machine type, using training and test sets of all available IDs.

3.2. Pre-Processing Phase

This section describes the pre-processing operations on the dataset for performing the experimental analysis, also discussing parameters selections regarding mel-spectrograms extraction, normalization, frames generation and IDs pre-processing. In particular, the same parameters are used for all machines in the mel-spectrograms extraction task, that has been performed by using the *Librosa* library²: the number of bins (n_mels) is 128, the STFT window (n_fft) is 1024 and the hop_length is 512.

²<https://librosa.org/>

Due to the duration of each clip (10 seconds) and the above mentioned parameters, each mel-spectrogram has the dimension of 128×313 .

Finally, four models for each architecture type must be trained to detect eventual anomalies. Moreover, for *match* and *not-match* transformations an $\alpha = 0.75$ is chosen, while the vector C is chosen equal to 5, after optimization.

3.3. Autoencoder Structure

This section outlines the structural design of the IDCCAE and IDC-LSTM-AE. The conditioning phase is a sequence of mathematical operations involving the encoder output and the ID conditioning network output (as detailed in Section 2). Specifically, the Encoded ID is analyzed through a dense and activation layer to produce the ID conditioning network's first output, which is then multiplied with the encoder output. The second output is computed via a dense layer with the same input provided to the Encoded ID, and the final representation (decoder input) is obtained by adding the multiplication output to the second output.

The encoder network comprises of five hidden layers with convolutional filters ranging from 32 to 512. Each encoder block includes a convolutional layer followed by batch normalization and ReLU activation. The bottleneck encompasses a layer with 40 convolutional filters, which reduces the encoder feature maps to a 40-dimensional encoded representation of the input. The decoder network involves a fully-connected layer that reshapes its input to the encoder's last layer's shape. Furthermore, five ConvolutionTransposeBlocks mirror the encoder, where each block contains Conv2DTranspose layers, batch normalization layers, and ReLU activation functions. Conditioning operations are as explained previously.

The LSTM based autoencoder consists of an encoder with three LSTM layers (64,32 and 16 units) and a decoder which is the reversed version of the encoder with a RepeatVector layer. The input is seen as time-series of 32 timesteps, characterized by 128 frequency amplitude features.

3.4. Hyperparameters Tuning

In this section we discuss about the hyperparameter tuning strategy of the proposed model. In particular, we explore the following model parameters: the constant vector C , which must be reconstructed by the autoencoder when the provided ID is wrong, α , being the percentage of correct frame-ID couples in training set. We performed a grid search setting α and C to $\{0.9, 0.75, 0.5\}$ and $\{0, 2.5, 5, 10\}$, respectively. During the training phase, other parameters are optimized for improving the performances of the proposed methodology. In addition to those seen for *IDCCAE* and *IDC-LSTM-AE*, we also optimized batch size ($\{64, 128, 256, 512\}$), number of epochs (in the $[50, 200]$ range) and learning rate ($\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$) as autoencoder-independent hyperparameters using ADAM as optimizer.

Finally, Mean Squared Error (MSE) has been chosen to evaluate reconstruction errors for all models and all machine types.

3.5. Evaluation and Performance Metrics

In this section are described the metrics used to evaluate the performances of trained models. The metrics used for models evaluation are the area under the receiver operating characteristic

Model	Pump		Fan		Valve		Slider		Memory (MiB)
	Training	Inference	Training	Inference	Training	Inference	Training	Inference	
<i>IDCCAE</i>	1h 26min 51s	3,19s	1h 58min 29s	3,04s	1 h 21 min 51s	2,94s	2h 2min 40s	2,78s	8
<i>IDC-LSTM-AE</i>	3min 49s	15,4s	6min 44s	14,7s	5 min 9s	15,1s	5min 56s	14,6s	64

Table 1

Efficiency evaluation of *IDCCAE* and *IDC-LSTM-AE* models with respect to *IDCAE* ones in terms of training and inference time and used memory.

(ROC) curve (AUC) and the partial-AUC (pAUC). The ROC curve shows the trend of the true positive rate (TPR) in function of the false positive rate (FPR) at the variation of a parameter, the pAUC is calculated as the AUC over a low FPR range $[0, p]$, with $p = 0.1$.

The anomaly score associated to a test sample is calculated taking the reconstruction errors average over all frames extracted from it and after the application of normalization. The pAUC is defined because it is especially important to increase the TPR under low FPR conditions, in that if an ASD system gives false alerts frequently we cannot trust it.

4. Results

This section discusses about the results obtained by the *IDCCAE* and *IDC-LSTM-AE* models with respect to different competitors in terms of efficiency and efficacy analysis.

We evaluated the efficiency of the proposed models by investigating their training and inference time, also considering their used memory. Specifically, we compare the two proposed models based on LSTM and CNN respectively.

Table 1 shows that the *IDC-LSTM-AE* achieves best results in terms of training although the inference time is highest, also requires a large amount of memory (64 MiB). On the other hand, *IDCCAE* requires a very high training time whilst achieving very good results in terms of both inference time (on average 2.98s) and used memory (8 MiB).

In turn, the efficacy analysis has been evaluated comparing the proposed models with respect to different competitors in terms of AUC and pAUC.

Table 2 shows results obtained by all models for each type of machinery. For pump machinery, *IDC-LSTM-AE* achieves the best result in terms of AUC (78.29%) while *IDCAE* [15] shows an increase of 0.66% in terms of pAUC w.r.t our approach. In turn, for fan machinery *IDCAE* and *IDCCAE* are the best models in terms of AUC and pAUC respectively (77.45% and 70.33%). In turn, for slider and valve *IDCCAE* achieves the highest results for pAUC metric 84.14%, while *CAE* results the best model for AUC metric reporting an increase of 0.78% and 4.1% w.r.t. *IDCCAE*.

Finally, the results show that the proposed methodology, which leverages *ID Conditioning*, *Mel-Spectrogram* and novel loss function for improving the model's performance, achieves the best value in terms of AUC and pAUC for all machines considered.

Model	Pump				Fan			
	AUC		pAUC		AUC		pAUC	
	Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev
Baseline	72.89%	0.70%	59.99%	0.77%	65.83%	0.53%	52.45%	0.21%
CAE [16]	72.07%	-	60.96%	-	66.78%	-	52.63%	-
LSTM [17]	73.94%	-	61.01%	-	67.32%	-	52.05%	-
IDCAE [15]	77.29%	-	70.33%	-	77.45%	-	70.32%	-
IDCCAE	76.63%	1.87%	67.90%	1.87%	71.05%	0.72%	70.33%	0.55%
IDC-LSTM-AE	78.29%	2.21%	69.67%	2.44%	67.66%	2.29%	65.83%	1.12%
Model	Slider				Valve			
	AUC		pAUC		AUC		pAUC	
	Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev
Baseline	84.76%	0.29%	66.53%	0.62%	66.28%	0.49%	50.98%	0.15%
CAE [16]	91.77%	-	76.20%	-	78.83%	-	53.10%	-
LSTM [17]	84.99%	-	67.47%	-	67.82%	-	51.07%	-
IDCAE [15]	80.04%	-	68.25%	-	78.26%	-	55.80%	-
IDCCAE	90.99%	4.30%	84.14%	6.46%	74.73%	5.00%	61.18%	5.07%
IDC-LSTM-AE	82.62%	1.90%	74.48%	2.64%	62.98%	2.99%	59.71%	1.53%

Table 2

Mean and std.dev. of AUC and pAUC for convolutional architectures on 10 independent trials. Results found in [16] are reported for comparison. Best results for each metric are marked in bold.

5. Discussion and Conclusions

Anomaly detection is a data-driven approach that employs predictive maintenance to minimize downtime, reduce costs, and optimize maintenance procedures. Numerous anomaly detection systems have been developed and studied in recent years. In this study, we propose a methodology for anomaly detection in predictive maintenance that incorporates machine identifiers into the autoencoder learning process. This method enables models to be trained on sounds recorded in the proximity of different versions of the same machine type, resulting in improved detection capabilities. We also employ algebraic operations to enhance the learning phase and leverage the latent representation of the input generated by the encoder. Our experiments, conducted on the DCASE 2020 Task 2 Challenge dataset, demonstrate that our approach improves performance by up to 17.61% compared to non-conditioned autoencoder versions and baseline models, particularly for pAUC.

Our future work will focus on analyzing various types of conditioning networks, such as *Variational AutoEncoders* (VAEs) or *Generative Adversarial Networks* (GANs), and applying different pre-processing strategies to achieve better training performance. These strategies include noise reduction to eliminate background noise commonly found in factory environments and audio data augmentation techniques such as pitching and time-shifting. Additionally, we plan to investigate two emerging research areas: Context Prediction and Context Histories, which utilize time series of Contexts to record machine data in context histories for different types of data analysis ([18, 19, 20]).

References

- [1] E. Di Fiore, A. Ferraro, A. Galli, V. Moscato, G. Sperli, An anomalous sound detection methodology for predictive maintenance, *Expert Systems with Applications* 209 (2022) 118324.
- [2] E. C. Nunes, Anomalous sound detection with machine learning: A systematic review, *arXiv preprint arXiv:2102.07820* (2021). doi:<https://doi.org/10.48550/arXiv.2102.07820>.
- [3] P. Calvo-Bascones, M. A. Sanz-Bobi, T. M. Welte, Anomaly detection method based on the deep knowledge behind behavior patterns in industrial components. application to a hydropower plant, *Computers in Industry* 125 (2021) 103376. doi:<https://doi.org/10.1016/j.compind.2020.103376>.
- [4] C. Yin, S. Zhang, J. Wang, N. N. Xiong, Anomaly detection based on convolutional recurrent autoencoder for iot time series, *IEEE Transactions on Systems, Man, and Cybernetics: Systems* (2020). doi:10.1109/TSMC.2020.2968516.
- [5] S. Thudumu, P. Branch, J. Jin, J. J. Singh, A comprehensive survey of anomaly detection techniques for high dimensional big data, *Journal of Big Data* 7 (2020) 1–30. doi:<https://doi.org/10.1186/s40537-020-00320-x>.
- [6] M. Cinque, R. Della Corte, V. Moscato, G. Sperli, A graph-based approach to detect unexplained sequences in a log, *Expert Systems with Applications* 171 (2021) 114556. doi:<https://doi.org/10.1016/j.eswa.2020.114556>.
- [7] L. De Simone, E. Caputo, M. Cinque, A. Galli, V. Moscato, S. Russo, G. Cesaro, V. Criscuolo, G. Giannini, Lstm-based failure prediction for railway rolling stock equipment, *Expert Systems with Applications* 222 (2023) 119767.
- [8] A. De Santo, A. Ferraro, A. Galli, V. Moscato, G. Sperli, Evaluating time series encoding techniques for predictive maintenance, *Expert Systems with Applications* 210 (2022) 118435.
- [9] A. Vafeiadis, K. Votis, D. Giakoumis, D. Tzouvaras, L. Chen, R. Hamzaoui, Audio content analysis for unobtrusive event detection in smart homes, *Engineering Applications of Artificial Intelligence* 89 (2020) 103226. doi:<https://doi.org/10.1016/j.engappai.2019.08.020>.
- [10] A. Ferraro, V. Moscato, G. Sperli, Deep learning-based community detection approach on multimedia social networks, *Applied Sciences* 11 (2021) 11447.
- [11] M. Meire, P. Karsmakers, Comparison of deep autoencoder architectures for real-time acoustic based anomaly detection in assets, in: *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, volume 2, 2019, pp. 786–790. doi:10.1109/IDAACS.2019.8924301.
- [12] B. Bayram, T. B. Duman, G. Ince, Real time detection of acoustic anomalies in industrial processes using sequential autoencoders, *Expert Systems* 38 (2021). doi:<https://doi.org/10.1111/exsy.12564>.
- [13] Y. Koizumi, S. Murata, N. Harada, S. Saito, H. Uematsu, Sniper: Few-shot learning for anomaly detection to minimize false-negative rate with ensured true-positive rate (2019) 915–919. doi:10.1109/ICASSP.2019.8683667.
- [14] H. Purohit, R. Tanabe, K. Ichige, T. Endo, Y. Nikaido, K. Suefusa, Y. Kawaguchi, Mimii

dataset: Sound dataset for malfunctioning industrial machine investigation and inspection, arXiv preprint arXiv:1909.09347 (2019). doi:<https://doi.org/10.48550/arXiv.1909.09347>.

- [15] S. Kapka, Id-conditioned auto-encoder for unsupervised anomaly detection, arXiv preprint arXiv:2007.05314 (2020). doi:<https://doi.org/10.5281/zenodo.4061782>.
- [16] A. Ribeiro, L. M. Matos, P. J. Pereira, E. C. Nunes, A. L. Ferreira, P. Cortez, A. Pilastrri, Deep dense and convolutional autoencoders for unsupervised anomaly detection in machine condition sounds, arXiv preprint arXiv:2006.10417 (2020). doi:<https://doi.org/10.48550/arXiv.2006.10417>.
- [17] A. Jalali, A. Schindler, B. Haslhofer, Dcase challenge 2020: Unsupervised anomalous sound detection of machinery with deep autoencoders (2020).
- [18] J. H. da Rosa, J. L. Barbosa, G. D. Ribeiro, Oracon: An adaptive model for context prediction, *Expert Systems with Applications* 45 (2016) 56–70. doi:<https://doi.org/10.1016/j.eswa.2015.09.016>.
- [19] V. La Gatta, V. Moscato, M. Postiglione, G. Sperli, Pastle: Pivot-aided space transformation for local explanations, *Pattern Recognition Letters* 149 (2021) 67–74. URL: <https://www.sciencedirect.com/science/article/pii/S0167865521002014>. doi:<https://doi.org/10.1016/j.patrec.2021.05.018>.
- [20] A. S. Filippetto, R. Lima, J. L. V. Barbosa, A risk prediction model for software project management based on similarity analysis of context histories, *Information and Software Technology* 131 (2021) 106497. doi:<https://doi.org/10.1016/j.infsof.2020.106497>.