# Deep Autoencoders for Acoustic Anomaly Detection: Experiments with Working Machine and In-Vehicle Audio

Gabriel Coelho[1], Luís Miguel Matos[1], Pedro José Pereira[1], André Ferreira[2], André Pilastri[3] and Paulo Cortez[1*]

[1]ALGORITMI Centre, Department of Information Systems, University of Minho, 4804-533 Guimarães, Portugal.
[2]Bosch Car Multimedia, 4705-820, Braga, Portugal.
[3]EPMQ - IT Engineering Maturity and Quality Lab, CCG ZGDV Institute, 4804-533 Guimarães, Portugal.

*Corresponding author(s). E-mail(s): pcortez@dsi.uminho.pt;
Contributing authors: b12159@algoritmi.uminho.pt; luis.matos@dsi.uminho.pt; pedro.pereira@dsi.uminho.pt; andre.ferreira2@pt.bosch.com; andre.pilastri@ccg.pt;

**Abstract**

The growing usage of digital microphones has generated an increased interest in the topic of Acoustic Anomaly Detection (AAD). Indeed, there are several real-world AAD application domains, including working machines and in-vehicle intelligence (the main target of this research project). This paper introduces three deep AutoEncoders (AE) for unsupervised AAD tasks, namely a Dense AE, a Convolutional Neural Network (CNN) AE and Long Short-Term Memory Autoencoder (LSTM) AE. To tune the deep learning architectures, development data was adopted from public domain audio datasets related with working machines. A large set of computational experiments was held, showing that the three proposed deep autoencoders, when combined with a melspectrogram sound preprocessing, are quite competitive and outperform a recently proposed AE baseline. Next, on a second experimental stage, aiming to address the final in-vehicle passenger safety goal, the three AEs were adapted to learn from in-vehicle normal audio, assuming three realistic scenarios that were generated by a

synthetic audio mixture tool. In general, a high quality AAD discrimination was obtained: working machine data – 72% to 91%; and in-vehicle audio – 78% to 81%. In conjunction with an automotive company, an in-vehicle AAD intelligent system prototype was further developed, aiming to test a selected model (LSTM AE) during a pilot demonstration event that targeted the cough anomaly. Interesting results were obtained, with the AAD system presenting a high cough classification accuracy (e.g., 100% for front seat locations).

**Keywords:** Acoustic anomaly detection, Unsupervised learning, Deep Autoencoders, Industrial and in-vehicle data, One-class learning.

# 1 Introduction

Due to advances in Information and Communications Technology (ICT) there is an increasing amount of physical processes that are collected in a digital form. In effect, currently there is a widespread usage of interconnected sensors that can capture diverse real-world phenomena (e.g., images, sound, temperatures). All this data can be used by Artificial Intelligence (AI) and Machine Learning (ML) to extract valuable analytics. A particularly relevant ML task is anomaly detection, which intends to distinguish abnormal events from normal ones [1, 2]. For instance, the early detection of operating machines with defects in industrial processes by using ML can potentially reduce maintenance time and costs, prevent or reduce production stops, and increase the safety of human operators that operate the machines [3, 4]. Moreover, anomaly detection by ML is valuable for future shared self-driving vehicles [5]. Given that there will be no human drivers, there is a need to automatically monitor the in-vehicle conditions in terms of the health, safety and comfort of its passengers. Thus, an intelligent system can analyze in-vehicle sensor data (e.g., images, sound) aiming to detect anomalies and potentially trigger actions (e.g., calling for assistance) [6].

The focus of this work is on ML methods for Acoustic Anomaly Detection (AAD) [7], which aims to detect abnormal behaviors using audio data. Several studies addressed this issue as an unsupervised ML task, since data labeling is highly costly and time consuming, requiring a large manual effort that is subject to errors [8]. Unsupervised AAD can be achieved by adopting an one-class learning [9, 10], by assuming ML algorithms such as Isolation Forest (IF) [11, 12] and One-Class Support Vector Machines (OC-SVM) [8, 13]. Moreover, following the success of Deep Learning, there has been a growing usage of neural network architectures for AAD. In particular, AutoEncoders (AE) are becoming popular for one-class AAD [14, 15]. When compared with other ML approaches (e.g., IF and OC-SVM), AE present the advantage of requiring a lower computational effort [16, 17], which is a valuable asset for designing AAD intelligent systems capable of working with operating machines and vehicles.

This work is set within a larger R&D project that addresses an unsupervised in-vehicle intelligence using multiple data sources (e.g., images, sound, particles) and that includes the *Bosch Car Multimedia S.A.* (BCM) company. This paper presents the full research related with the in-vehicle AAD component of the R&D project, which corresponds to a large extended version of a previously published conference paper [18]. The main goal is to detect in-vehicle AAD, such as passengers arguing, coughing or an accidental breaking of glass. As previously argued [6], audio data is very scarce within this domain and there are no public domain datasets. Thus, in an initial stage, in order to tune and compare different AE methods, a different application domain scenario was approached by assuming public data related with toy and real working machines [18], as provided by the ToyADMOS [19] and the MIMII [20] datasets. Following on good results obtained in previous studies [2, 21–23], three different AE architectures were adjusted and compared: deep Dense AE, Convolutional Neural Network (CNN) AE and Long Short-Term Memory Autoencoder (LSTM) AE. During the machine sound experiments, the popular Mel Frequency Energy Coefficients (MFECs) [24, 25] were adopted to preprocess the audio. For benchmark purposes, the obtained results were compared with a baseline AE architecture that was recently proposed [26]. Next, on a second stage the R&D project in-vehicle scenario was targeted. In a previous conference paper [6], a synthetic in-vehicle sound simulator was developed, aiming to perform a realistic mixture of abnormal and normal audio clips with normal car driving background sound. To evaluate the usefulness of the generated in-vehicle data, the same work [6] performed an initial comparison study using: two sound preprocessing methods, MFECs and a combination of three features (e.g., Mel Frequency Cepstral Coefficients); and two AE architectures, a deep Dense AE and LSTM AE. This paper uses the three synthetic datasets that were created by the in-vehicle sound simulator and it assumes the MFEC sound preprocessing method that obtained better results in both [18] and [6] studies. A total of three AE AAD methods are compared (Dense AE, LSTM AE and CNN AE). When compared with [6], the proposed AEs include an improved LSTM AE architecture (e.g, with Batch Normalization layers) and a novel CNN AE. Moreover, this paper presents new results regarding a pilot demonstration experiment that was conducted at BCM, involving a selected AAD method (LSTM AE), the cough abnormal use case and a two-day human testing (e.g., involving business managers, journalists and students).

The paper is organized as follows: section 2 describes the sound datasets, the audio preprocessing, the proposed AAD AE architectures and the evaluation process. Section 3 presents the experiments conducted and and obtained results. Finally, the main conclusions are discussed in section 4.

# 2 Materials and methods

## 2.1 Audio data

This work considers several unsupervised learning experiments related with two application domain datasets. One considers an industrial environment and the other consists of in-vehicle audio data. Each domain data was originally recorded using a distinct Sample Rate ($SR$): 16,000 Hz for the sound machines and 44,100 Hz for the in-vehicle audio. It should be noted that the AAD methods do not work directly with the raw data but with a preprocessed Fast Fourier Transformation (FFT) that is associated with a reduced input dimension (see section 2.2).

The industrial data is related with parts of the ToyADMOS [19] and the MIMII [20] datasets, consisting of the normal and anomalous operating sounds of six types of toy/real machines, as obtained from the Detection and Classification of Acoustic Scenes and Events (DCASE) 2020 challenge [26]. The data is divided into two parts (development and evaluation) for 6 different machine types: ToyCar, ToyConveyor, Slider, Pump, Fan, and Valve.

The ToyCar and ToyConveyor data belong to ToyADMOS dataset. This dataset involves miniature machines (toys) that were damaged deliberately to record anomalous behavior. As for the MIMII Dataset, the sounds are recorded from different industrial machines, aiming to resemble a real-life scenario. In the development datasets, each machine type has 4 different specific machines, except for ToyConveyor, which has only 3.

The machine sound datasets include normal and anomalous labels that are available for the test data, allowing to estimate the AAD performance of the ML models. Regarding the evaluation data, it contains audio for new machines (new IDs) in each machine type, both for model training and testing. Table 1 summarizes the analyzed datasets for the industrial application domain. It should be noted that a different number of approximately 10 second Waveform Audio File (WAV) files is used for each machine (column Test).

Regarding the in-vehicle audio, it is related with a synthetic sound simulator that was developed using the Python language [6]. The simulator works as a sound mixture tool that uses background recordings (194 sound files) and acoustic scenes relative to a set of predefined normal and abnormal behaviors (short audio clips). Each generated synthetic audio file had a duration of 60 seconds and it was randomly sampled from the whole raw car driving audio data. The background clips were manually labeled, allowing to identify two relevant subclasses: radio on - 160 clips; and radio off - 34 clips. The rare events were related with audio clips with a few seconds that were collected from two public sources: Freesound[1] search engine and Librivox[2]. The collected audio clips were associated with the 7 use cases from the R&D project: **anomaly events** - people arguing, breaking a window or coughing; and **normal events** - reading a book, singing, talking or using a smartphone (e.g.,

[1] https://freesound.org/
[2] https://librivox.org/

**Table 1** Summary of the working machine AAD datasets.

| | Development | | | Evaluation | | |
|---|---|---|---|---|---|---|
| | Machine | Audio Files | | Machine | Audio Files | |
| | ID | Train | Test | ID | Train | Test |
| Toy Car | 01 | 1000 | 614 | 05 | 1000 | 515 |
| | 02 | 1000 | 615 | 06 | 1000 | 515 |
| | 03 | 1000 | 615 | 07 | 100 | 515 |
| | 04 | 1000 | 615 | | | |
| Toy Conveyor | 01 | 1000 | 1200 | 04 | 1000 | 555 |
| | 02 | 1000 | 1155 | 05 | 1000 | 555 |
| | 03 | 1000 | 1154 | 06 | 1000 | 555 |
| Fan | 00 | 911 | 507 | 01 | 911 | 426 |
| | 02 | 916 | 549 | 03 | 916 | 458 |
| | 04 | 933 | 448 | 05 | 1000 | 458 |
| | 06 | 915 | 461 | | | |
| Pump | 00 | 906 | 243 | 01 | 903 | 2016 |
| | 02 | 905 | 211 | 03 | 606 | 213 |
| | 04 | 602 | 200 | 05 | 908 | 348 |
| | 06 | 936 | 202 | | | |
| Slider | 00 | 968 | 456 | 01 | 968 | 278 |
| | 02 | 968 | 367 | 03 | 968 | 278 |
| | 04 | 434 | 278 | 05 | 434 | 278 |
| | 06 | 434 | 189 | | | |
| Valve | 00 | 891 | 219 | 01 | 679 | 220 |
| | 02 | 608 | 220 | 03 | 863 | 220 |
| | 04 | 900 | 220 | 05 | 899 | 500 |
| | 06 | 892 | 220 | | | |

texting). Table 2 summarizes the retrieved audio events: the type of **Use case**, if it is an **Anomaly** and the total number of audio clips (column **Clips**). In the table, the last three columns are related with the three generated AAD datasets: Mix3 – includes radio on sound and assumes a smaller in-vehicle size; Mix6 – similar to Mix3 but assumes a larger vehicle interior size (e.g., van); Mix6NR – similar to Mix6 but does not include radio on background sound (NR stands for "No Radio"). For the three datasets, the training and test sizes are shown in seconds.

## 2.2 Audio feature extraction

This research assumes a spectral analysis to extract features from the audio signals, which is a popular approach to preprocess audio [27]. In particular, a feature for audio signal processing named Mel Frequency Energy Coefficients

**Table 2** Summary of the in-vehicle AAD datasets created in [6].

| Use case | Anomaly | Clips | Split | Class | Mix3 (s) | Mix6 (s) | Mix6NR (s) |
|---|---|---|---|---|---|---|---|
| Background | | 194 | train | normal | 4200 | 4200 | 4200 |
| | | | test | normal | 1800 | 1800 | 1800 |
| Arguing | ✓ | 32 | train | normal | 3443 | 3469 | 3521 |
| | | | | anomaly | 757 | 731 | 679 |
| | | | test | normal | 1460 | 1434 | 1470 |
| | | | | anomaly | 340 | 366 | 330 |
| Breaking Window | ✓ | 64 | train | normal | 3572 | 3577 | 3787 |
| | | | | anomaly | 628 | 623 | 413 |
| | | | test | normal | 1546 | 1541 | 1530 |
| | | | | anomaly | 254 | 259 | 270 |
| Cough | ✓ | 49 | train | normal | 3774 | 3807 | 3888 |
| | | | | anomaly | 426 | 393 | 312 |
| | | | test | normal | 1671 | 1638 | 1670 |
| | | | | anomaly | 129 | 162 | 130 |
| Reading | | 67 | train | normal | 4200 | 4200 | 4200 |
| | | | test | normal | 1800 | 1800 | 1800 |
| Singing | | 47 | train | normal | 4200 | 4200 | 4200 |
| | | | test | normal | 1800 | 1800 | 1800 |
| Talking | | 71 | train | normal | 4200 | 4200 | 4200 |
| | | | test | normal | 1800 | 1800 | 1800 |
| Using Smartphone | | 13 | train | normal | 4200 | 4200 | 4200 |
| | | | test | normal | 1800 | 1800 | 1800 |

(MFECs) is addressed, which are log-energies derived directly from the filter-banks energies. This feature provided good results in detecting different audio sounds and classification of sounds in previous studies [22, 28, 29]. For demonstration purposes, fig. 1 exemplifies the original sound waveforms for normal and abnormal Fan machine events, while fig. 2 displays the same sound events when using the MFEC representation. Visually, it becomes clear that there are more signal differences between normal and abnormal events when adopting the MFEC transformation.

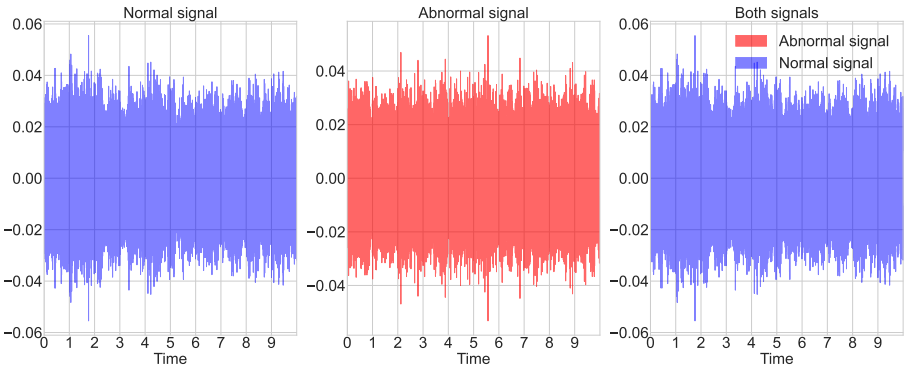


**Fig. 1** Examples of machine Fan ID 00 sound waveforms.

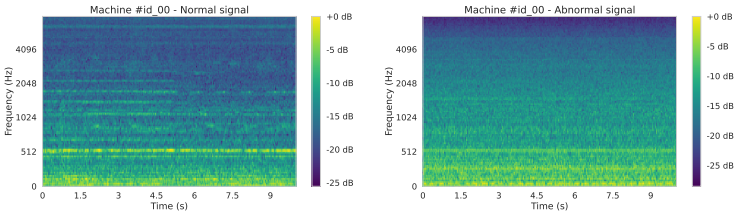


**Fig. 2** Mel Scale representation for machine Fan ID 00 normal and abnormal examples using the energy values (MFECS).

To prepare the features for two of the explored deep learning architectures, namely the Dense and LSTM AE, some operations were made. In this research, and similarly to the DCASE competition [26], the two AEs work with sound samples with a length of one second (for training and testing). Thus, Audio data were buffered in fixed-length 1-second intervals with a 50% overlap. For each audio buffer obtained, the segment was then divided into $F = \frac{NFFT}{SR}$ ms analysis Frames ($F$), with 50% overlap, with 128 MFECs being extracted from the magnitude spectrum of each frame. The $NFFT$ denotes the length of the FFT window and $SR$ denotes the sound sample rate (in Hz). In this work, the $NFFT$ window length is fixed to 1024, while the $SR$ is application domain dependent (16,000 Hz for the working machines and 44,100 Hz for the in-vehicle audio). In this way, $NF$ =5 time-frames are concatenated to form a 640-dimensional input vector for each sound second segment, as shown in fig. 3. The `JAX` python module [30] was used to implement this sound pre-processing procedure. The module uses the GPU cores to read, process and extract MFECs directly from the sound waveforms.

As for the Convolution Neural Network (CNN) AE, it requires a slightly different feature extraction method. The CNN model mainly uses images as an input for the model and it requires much more computation than the other AEs. Therefore, in order to reduce the computational effort, a different pre-processing approach is assumed, in which only 128 MFECs are extracted for a complete training audio file (with several seconds) by considering $F = \frac{NFFT}{SR}$ ms frames with a 50% overlap. This generates one image per audio file of size $NF \times 128$ as shown in fig. 4. The $NF$ parameter denotes the total number of frames, each with a $F$ time length, for a particular sound file size and when assuming a 50% overlap. In order to perform the same test comparison that was executed for the other AEs, during the testing phase the test audio files are first divided into shorter one second segments, thus computing a distinct MFEC image for each segment.

## 2.3 Autoencoder Architectures

AEs compress numeric input features into a lower dimensional space, named latent space, and then output back the original input signal. An AE is composed of two main components, an encoder that computes the latent space,

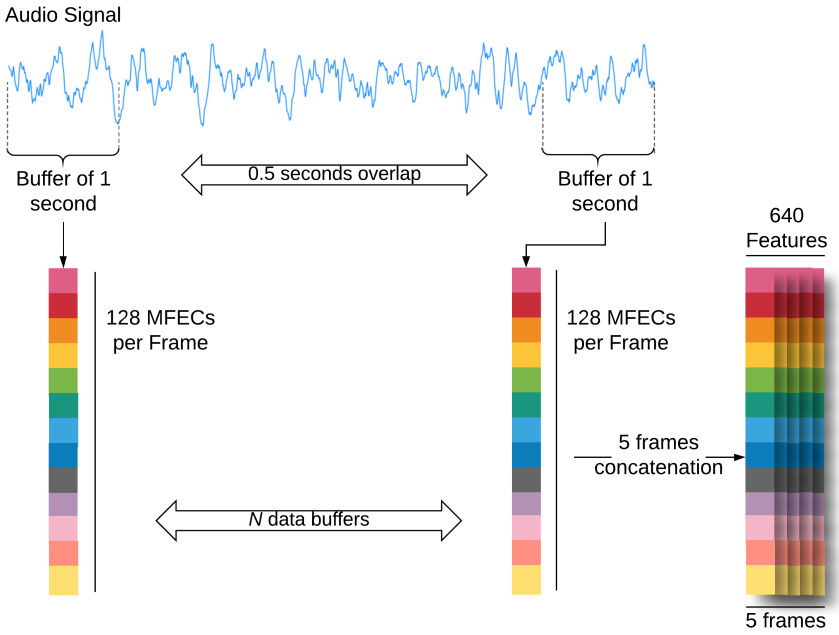


**Fig. 3** Feature extraction procedure for the Dense and LSTM AE.

via a nonlinear transformation, and a decoder that attempts to reconstruct the reverse transformation [31, 32]. Thus, the AE performs a regression task, aiming to minimize the reconstruction error, which is defined as the difference between the original numeric input vector and the AE output [33]. It is often assumed that normal and anomalous events follow different distributions, thus AEs are typically trained to learn the normal multi-dimensional space of the data by using only normal event records. The reconstruction error can then be used to detect class anomalies by assuming a threshold value (Th), such that an anomaly is considered positive if the reconstruction error is higher than the threshold [31, 32].

All proposed AEs are trained and tested with MFEC processed sound segments (section 2.2), each with a one second length. In order to select the best AE architectures, several preliminary experiments were conducted by considering only development data from the first application domain data (working machine datasets). The best configuration (in terms of the reconstruction error) was selected by varying elements such as the number of hidden layers and units per layer. Once the neural architecture was selected, it was fixed and applied to all datasets. For both AEs, the training only uses normal event sounds.

The first proposed architecture consists of a deep fully-connected (thus Dense) AE (top of fig. 5), which was adopted in the baseline AE proposed in [26]. The best preliminary results were achieved by a Dense AE that includes

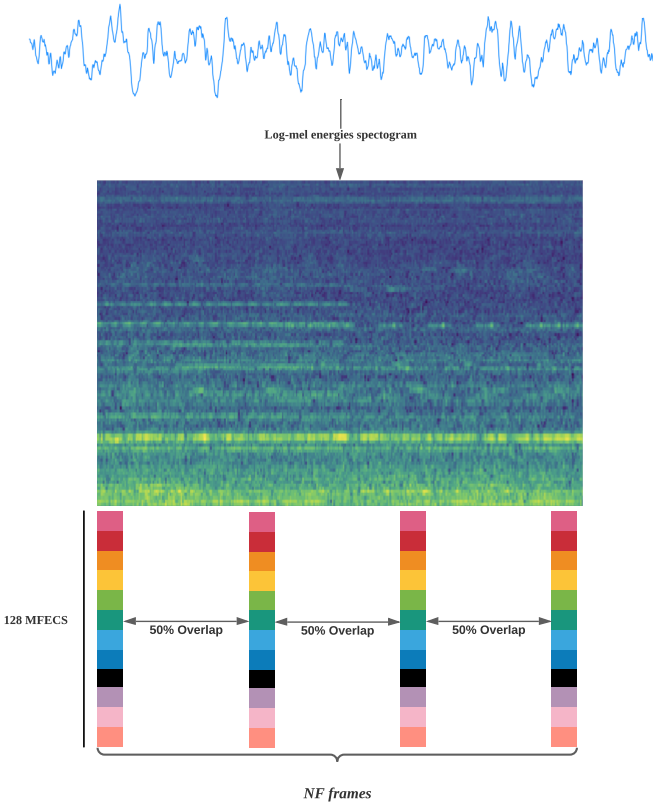


**Fig. 4** Feature extraction procedure for the CNN AE.

encoder and decoder components with four fully-connected layers with 512 hidden units, followed by a Batch Normalization, with all neural nodes using the popular ReLU as activation function. The Batch Normalization layer allows reduce the internal covariate shift, discarding the need of dropout, and normalizes the inputs for each batch of data [34]. As for ReLU, it presents the advantage of non-saturation of its gradient, which greatly accelerates the convergence of stochastic gradient descent compared to other activation functions, including logistic or hiperbolic tangent [35]. The bottleneck layer is set as one fully-connected layer with 8 hidden units, resulting in an 8-dimensional latent space.

Recently, CNNs have achieved promising results on many AAD benchmarks [36–38]. By integrating 2D convolutional operations in an AE structure, CNN AEs are capable of learning the spatial structure of the input features and reconstruct them while taking into account their spatial structural patterns. Based on this property, the second proposed deep learning architecture for the unsupervised AAD task consists of a deep CNN AE (shown in the

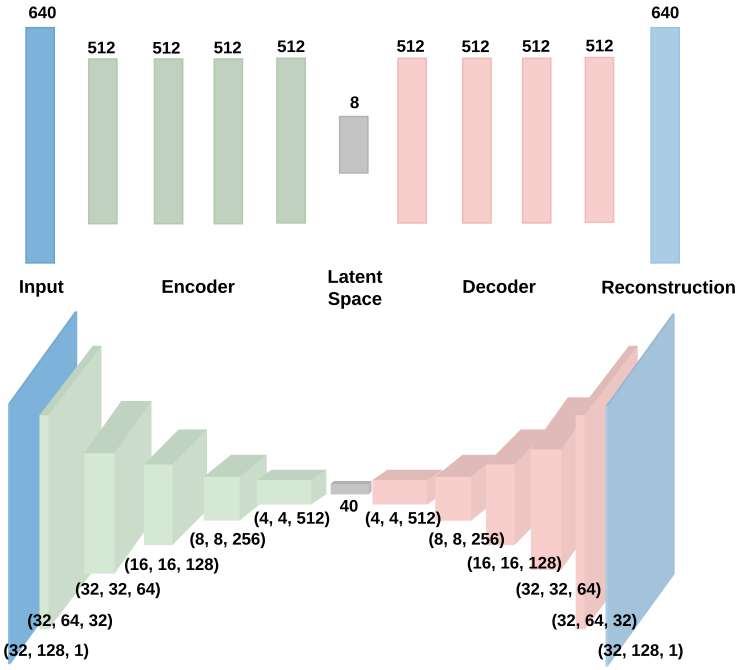


**Fig. 5** Proposed AE Network architectures: Dense AE (top) and CNN AE (bottom).

bottom of fig. 5). With such an architecture, the AAD task is handled as a computer vision problem by exploring image-like time-frequency representations of audio. The encoder and decoder networks are comprised of convolutional blocks, each consisting of 2D Convolution and Batch Normalization layers, using ReLU as the activation function. The encoder network is composed by a stack of five convolutional layers with 32, 64, 128, 256, and 512 convolutional filters, kernel sizes of 5, 5, 5, 3, and 3 to capture local patterns, and strides of $(1, 2)$, $(1, 2)$, $(2, 2)$, $(2, 2)$, and $(2, 2)$, respectively. The feature size map is reduced throughout the encoder by the convolution operation stride. The bottleneck consists of a layer with 40 convolutional filters, reducing the encoder feature maps to a 40-dimensional compressed input representation. Concerning the decoder network, it starts from the bottleneck layer (the latent space) and then it includes several hidden layers, which attempt to reconstruct the latent space representation back into the original input. To avoid computational memory issues, during the training of the CNN AE, the data generator of the Keras Python module is adopted, which feeds just one image at the time into the CNN AE.

The LSTM is a special type of deep Recurrent Neural Network (RNN) that solves the "short-term memory" problem through a mechanism of gates

that manage to regulate the flow of information [39, 40]. Since the LSTM neural network tends to provide good results when modeling temporal data, this work proposes an LSTM-AE architecture for AAD tasks, as presented in fig. 6. In this AE, the encoder and decoder components are composed of LSTM layers. Similarly to the simpler AE, each component is composed of a stack of five LSTM layers with $L_h$ cells followed by a Batch Normalization ($BN$) layer per stack. As for the activation function for each LSTM layer, it consists of the Exponential Linear Unit (ELU) activation function that saturates better for negative net inputs and diminishes the vanishing gradient issue, therefore enabling a faster learning [41]. The LSTM architectures tend to produce exploding gradients given the accumulation of gradients unrolled over hundreds of input time steps. To solve this issue, Batch Normalization (BN) layers were adopted, as suggested in [42]. There is also a bottleneck layer with $L_b$ hidden cells, where $L_b < L_h$. This layer is preceded by a LSTM layer with the same size ($L_b$), which acts as a bridge, preparing the data to be decoded. Since the decoder network is designed to unfold the encoding component, the decoder layers were stacked in the reverse order of the encoder ones. The selected number of hidden nodes per layer is shown in fig. 6 (e.g., $L_h$ =128, $L_b$ =8).

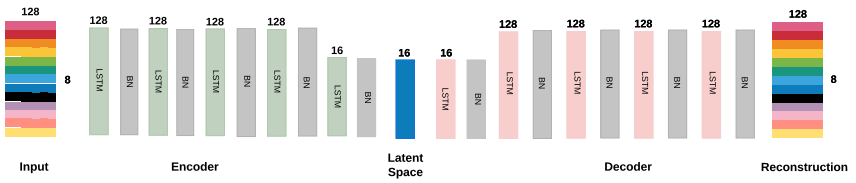


**Fig. 6** Proposed LSTM structure.

Regarding the training algorithm used to train AE architectures, it consists of the Adam optimizer, which also was used in [26], using a learning rate of 0.001. All three AE types were trained to minimize the Mean Squared Error (MSE) between input and its reconstruction (the loss function). The training procedure was iterated up to a maximum of 100 epochs. In each epoch, 10% of training data was randomly divided for validation purposes, allowing to monitor the evolution of the reconstruction error. If the MSE does not improve on validation data after 10 epochs, an early stopping is activated, ending the training process and storing the weights of the model that achieved a lower reconstruction error on validation data. As for the batch size, for both Dense AE and LSTM AE architectures it was set to 512, while for the CNN AE architecture it was set to 64.

Once the AE is trained, the MSE reconstruction error for an unseen sound sample $i$ (related with a short length of one second) is used as the decision

score ($d_i$):

$$MSE_i = \frac{\sum_{j=1}^{I}(x_{i,j} - \hat{x}_{i,j})^2}{I} \quad (1)$$

where $x_{i,j}$ and $\hat{x}_{i,j}$ denote the desired input value and estimated AE output for the $i$-th data instance and $j$-th input or output node and $I$ denotes the total number of the AE inputs (and outputs). The reconstruction MSE error is used as the decision score $d_i = \text{MSE}_i$, where higher reconstruction errors should correspond to a higher anomaly probability.

### 2.4 Evaluation

Given that a binary classification task is approached, the predictive performance is evaluated by adopting two popular AAD metrics that are based on the Receiver Operating Characteristic (ROC) analysis, namely the Area Under the ROC Curve (AUC) and partial-AUC (pAUC) [19, 20]. The ROC curve shows the False Positive Rate (FPR) versus the True Positive Rate (TPR) for different threshold values (Th). In this research, the positive class is the anomaly.

The AUC measure represents the overall ML discrimination performance, while pAUC focuses on a particular range of interest from the ROC curve, defined in this work as the FPR values from 0 to 0.1, which reflects in a model with fewer false alarms. Quality values for both measures are not influenced by unbalanced data, which in occurs in the AAD datasets. The AUC and pAUC values can be interpreted as follows: 50% performance of a random classifier; 60% - reasonable; 70% - good; 80% - very good; 90% - excellent; and 100% - perfect [43].

As mentioned in section 2.3, the three AE architectures were set by using development data from the working machine application domain datasets. A single AE model was fit (using training data) and then evaluated (using test data) for each operating machine type (e.g., ID 04 for Toy Car) and in-vehicle mixture dataset (e.g., Mix3).

## 3 Results

The proposed AE architectures were implemented in the Python programming language, using the TensorFlow-GPU library [44]. The computational experiments were conducted using two different GPUs (Titan Xp and 1080Ti). To evaluate the model performance, both AUC and pAUC metrics were used, as defined in section 2.4.

### 3.1 Working machine audio

Table 3 presents the obtained predictive results for each specific machine, also showing the average value for each machine type. For comparison purposes, the Baseline system results from [26] are also provided in the table. The obtained

**Table 3** Comparison of AUC and pAUC results for all AE architectures for each machine (best average values are denoted in **bold**)

| Machine Type | Machine ID | Baseline | | Dense AE | | CNN AE | | LSTM AE | |
|---|---|---|---|---|---|---|---|---|---|
| | | AUC (%) | pAUC (%) | AUC (%) | pAUC (%) | AUC (%) | pAUC (%) | AUC (%) | pAUC (%) |
| ToyCar | 1 | 81.36 | 68.40 | 83.87 | 72.64 | 81.59 | 71.88 | 80.97 | 66.67 |
| | 2 | 85.97 | 77.72 | 87.56 | 80.35 | 85.46 | 79.92 | 85.56 | 68.89 |
| | 3 | 63.30 | 55.21 | 63.12 | 55.02 | 62.73 | 55.08 | 68.43 | 58.13 |
| | 4 | 84.45 | 68.97 | 88.60 | 76.68 | 82.38 | 69.60 | 87.16 | 73.02 |
| | **Average** | 78.77 | 67.58 | **80.79** | **71.17** | 78.04 | 69.12 | 80.53 | 66.68 |
| ToyConveyor | 1 | 78.07 | 64.25 | 81.67 | 69.41 | 79.90 | 62.71 | 77.10 | 60.82 |
| | 2 | 64.16 | 56.01 | 68.04 | 58.31 | 67.78 | 54.85 | 64.09 | 54.84 |
| | 3 | 75.35 | 61.03 | 79.59 | 63.64 | 80.11 | 62.53 | 80.02 | 63.50 |
| | **Average** | 72.53 | 60.43 | **76.43** | **63.79** | 75.93 | 60.03 | 73.74 | 59.72 |
| fan | 0 | 54.41 | 49.37 | 56.73 | 49.72 | 51.77 | 49.05 | 52.53 | 49.14 |
| | 2 | 73.40 | 54.81 | 79.60 | 54.00 | 72.71 | 55.51 | 79.72 | 59.03 |
| | 4 | 61.61 | 53.26 | 70.11 | 54.11 | 62.60 | 52.80 | 61.35 | 52.06 |
| | 6 | 73.92 | 52.35 | 81.69 | 55.15 | 80.05 | 53.19 | 86.90 | 58.65 |
| | **Average** | 65.83 | 52.45 | **72.03** | **53.25** | 66.78 | 52.63 | 70.13 | 54.72 |
| pump | 0 | 67.15 | 56.74 | 66.94 | 56.83 | 66.37 | 54.95 | 66.09 | 54.99 |
| | 2 | 61.53 | 58.10 | 60.77 | 60.31 | 54.31 | 53.58 | 60.54 | 58.51 |
| | 4 | 88.33 | 67.10 | 87.00 | 66.32 | 94.64 | 77.26 | 97.15 | 86.32 |
| | 6 | 74.55 | 58.02 | 77.53 | 60.32 | 76.97 | 58.05 | 76.00 | 56.35 |
| | **Average** | 72.89 | 59.99 | 73.06 | 60.94 | 72.07 | 60.96 | **74.95** | **64.04** |
| slider | 0 | 96.19 | 81.44 | 96.12 | 82.30 | 98.86 | 94.47 | 96.23 | 81.10 |
| | 2 | 78.97 | 63.68 | 79.55 | 64.42 | 84.06 | 69.33 | 82.41 | 61.94 |
| | 4 | 94.30 | 71.98 | 95.44 | 76.14 | 97.69 | 87.82 | 92.98 | 65.38 |
| | 6 | 69.59 | 49.02 | 77.22 | 49.56 | 86.46 | 53.16 | 75.93 | 49.32 |
| | **Average** | 84.76 | 66.53 | 87.08 | 68.10 | **91.77** | **76.20** | 86.89 | 64.44 |
| valve | 0 | 68.76 | 51.70 | 74.61 | 52.28 | 78.69 | 52.59 | 81.10 | 53.91 |
| | 2 | 68.18 | 51.83 | 76.68 | 52.72 | 85.02 | 55.92 | 78.98 | 53.11 |
| | 4 | 74.30 | 51.97 | 79.58 | 50.96 | 82.59 | 53.68 | 81.58 | 52.46 |
| | 6 | 53.90 | 48.43 | 57.78 | 48.73 | 69.03 | 50.22 | 66.76 | 50.04 |
| | **Average** | 66.28 | 50.98 | 72.16 | 51.17 | **78.83** | **53.10** | 77.11 | 52.38 |

results show that the proposed three AEs tend to outperform the Baseline system, which achieved best results in just 2 of the 23 analyzed specific machines (for the pump machine IDs 0 and 2). As for the comparison between the three AEs, there is no clear winner. The Dense AE obtained the best average values (for both AUC and pAUC measures), regarding the Toycar, Toyconveyor and fan machines. Turning to the CNN model, the best AAD results values were obtained for the slider and valve machines. Regarding the LSTM AE architecture, the fitted models obtained the best average AAD predictive performance for the pump machine. Overall, it should be noted that when considering the AUC measure, a high quality anomaly class discrimination was achieved by the proposed AEs, since most AUC values are above 70%.

To further demonstrate the quality of the obtained results, the left graphs of fig. 7 exemplify some of the ROC curves related with the machine sound AAD results (table 3). For each curve, two distinct thresholds were selected, allowing to compute two confusion matrices, as shown in the right plots of fig. 7. In the plots, the more sensitive and lower threshold (with its respective confusion matrix) is colored in green, while the more specific and higher threshold point is colored in red. From each confusion matrix, it is possible to compute several classification measures [45], such as the known Accuracy, Precision and Recall statistics that are presented in table 4.

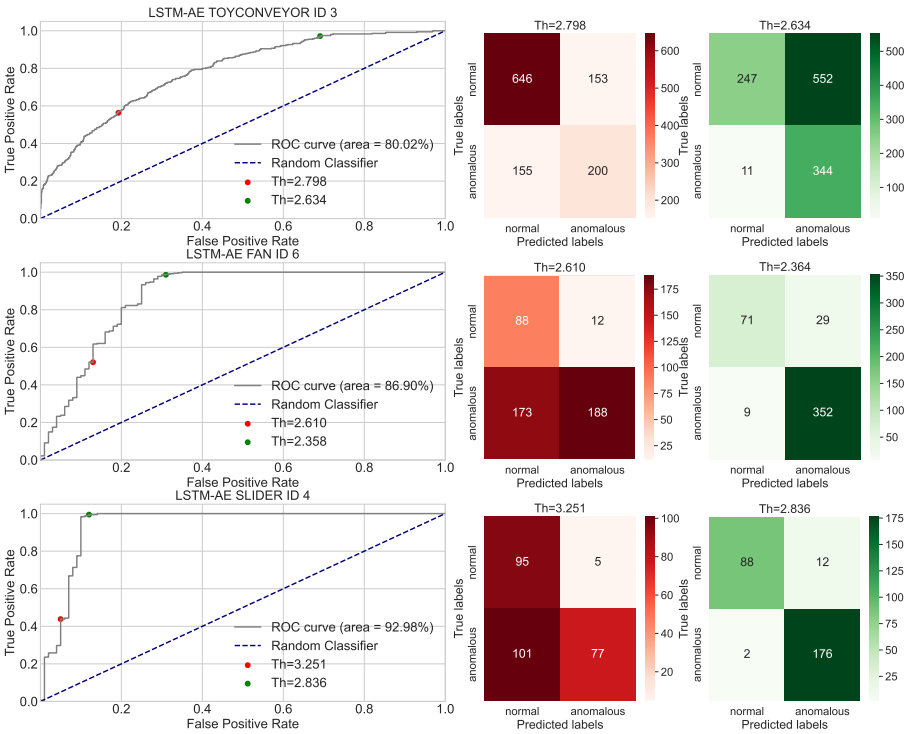


**Fig. 7** Machine sound ADD examples of ROC curves (left graphs) and confusion matrices related with two thresholds (right plots).

**Table 4** Class label prediction results for the threshold example values from fig. 7.

| | | Classification measure | | |
|---|---|---|---|---|
| **Model** | Th | Accuracy | Precision | Recall |
| LSTM-AE ToyConveyor ID 3 | 2.634 | 51.2% | 38.4% | 96.9% |
| | 2.798 | 73.3% | 56.7% | 56.3% |
| LSTM-AE Fan ID 6 | 2.364 | 92.4% | 92.2% | 98.6% |
| | 2.610 | 59.9% | 94.0% | 52.1% |
| LSTM-AE Slider ID4 | 2.836 | 95.0% | 93.6% | 98.9% |
| | 3.251 | 61.9% | 93.9% | 43.3% |

## 3.2 In-vehicle audio

After evaluating the performance of the three AEs that were tuned using working machine data, the same architectures were explored for the targeted in-vehicle domain. Given the need to design a real working intelligent AAD system, the computational effort required by the three AE architectures when assuming the same server (Titan Xp GPU) was also recorded. The obtained results are presented in table 5 and include the training (**Train**, in s) and

prediction (**Predict**) times (in ms). It should be noted that the previously Baseline method is not included in this table because it was specifically developed for the working machine audio and this learning method is not directly available, only the machine AAD predictions. An analysis of table 5 shows

**Table 5** Comparison of AUC and pAUC results for all AE architectures for each in-vehicle dataset (best average values are highlighted in **bold**).

| Audio Mixture | AAD Model | Train Time (s) | Predict Time (ms) | AUC | pAUC |
|---|---|---|---|---|---|
| Mix3 | Dense AE | 4861 | 0.7 | 73.08 | 56.87 |
| | LSTM AE | 8243 | 25.7 | 77.27 | 60.29 |
| | CNN AE | 4565 | 53.13 | **78.14** | **65.71** |
| Mix6 | Dense AE | 3564 | 0.8 | 72.21 | 56.96 |
| | LSTM AE | 8015 | 25.7 | **78.27** | 60.81 |
| | CNN AE | 19605 | 55.83 | 77.24 | **62.19** |
| Mix6NR | Dense AE | 1010 | 6.0 | 77.56 | 57.38 |
| | LSTM AE | 8293 | 31.1 | 78.54 | 58.57 |
| | CNN AE | 5009 | 53.23 | **81.34** | **64.69** |

that the CNN AE obtained the best AAD predictive performance (when considering both AUC and pAUC measures) for the Mix3 and Mix6NR scenarios. The LSTM AE outperformed the Dense and LSTM AEs in terms of the AUC values for the Mix6 dataset. As for the Dense AE, this deep learning architecture obtained the worst AUC and pAUC values for the three tested in-vehicle datasets. Overall, it should be noted that a high quality AAD discrimination was achieved, with the best AEs producing an AUC values of 78% (Mix3 and Mix6) and 81% (Mix6NR). Regarding the computational times, the Dense AE is in general the fastest method, tending to require less training and inference (prediction) effort. The LSTM AE architecture requires a stable training effort for all three datasets (around 8,000 s) and the second fastest inference (predict) time. The CNN AE requires the highest training time for the Mix6 dataset. Moreover, it also presents the highest AAD inference time for all in-vehicle datasets. For demonstrative purposes, fig. 8 presents the full ROC curve for the Mix 6 LSTM AE model and two confusion matrices associated with a more sensitive (green color) and a more specific (red color) threshold values.

The in-vehicle results were shown to the BCM company, which provided a very positive feedback. In addition to the computational experiments carried out, there was the opportunity to test one algorithm in a real environment, as part of a pilot event promoted by BCM. The microphone used for this test was a *Mic Array UMA-8-SP*, which is a multi-channel USB microphone paired with a digital audio amplifier. This sensor has seven high performance Micro Electro Mechanical System (MEMS) microphones configured in a circular arrangement to ensure high quality voice capture for heterogeneous types of applications. For this real-world experiment, the microphone was installed

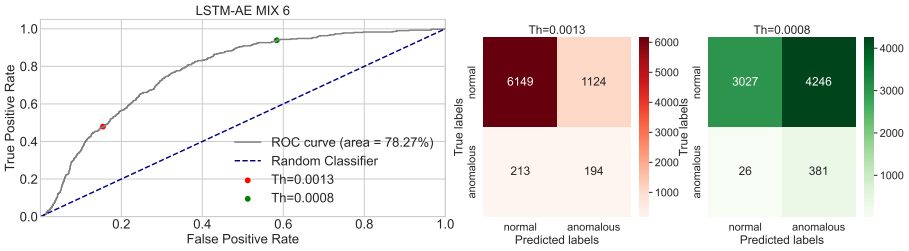


**Fig. 8** In-vehicle ADD example of a ROC curve (left graph) and confusion matrices related with two thresholds (right plot).

behind the rear-view mirror of the car. The left of fig. 9 shows a photograph of the microphone placement inside the vehicle used for the event demonstration, which was connected into a laptop that executed the AAD intelligent system (shown at the bottom of the same photograph).

**Fig. 9** Photographs of the in-vehicle AAD system (left) and its computational interface prototype (right) adopted during the BCM pilot demonstration.

Cough was the abnormal use case chosen by BCM to test the pilot, since this action is more easy to simulate by humans. Moreover, in contrast with the Breaking Window event, it does not impact physically in the in-vehicle

environment, allowing repeated cough executions for both single and multiple users. Given that the pilot real in-vehicle environment contained a higher level of ambient noise when compared with the synthetic in-vehicle datasets, the BCM company collected several hours of real audio data before the pilot demonstration and by using the demonstration vehicle under different conditions (e.g., stopped with engine, air conditioner and radio on and off). To select the ADD model, the Mix6 scenario was considered from the synthetic in-vehicle datasets, since it is more closely related with the expected BCM demonstration environment (e.g., large vehicle, possibly with radio on). Table 6 details the individual LSTM AE and CNN AE results (the best AAD architectures reported in table 5) for each abnormal use case (Cough, Argument, Breaking Window) when considering the Mix6 data. For the pilot use case (Cough), the LSTM AE obtained the highest AUC value (85%). Considering that LSTM AE model also requires less computational effort (in both training and prediction times, see table 5), it was the AE architecture selected for the event demonstration and that was further trained with the collected BCM audio.

**Table 6** LSTM AE and CNN AE subset test results for each anomaly use case when using the **Mix6** data.

| Use Case | AAD Model | AUC | pAUC |
|---|---|---|---|
| **Cough** | LSTM-AE | **85.15** | 71.85 |
| | CNN-AE | 83.37 | **72.69** |
| Argument | LSTM-AE | 88.89 | 77.59 |
| | CNN-AE | 93.44 | 83.56 |
| Breaking Window | LSTM-AE | 71.15 | 53.01 |
| | CNN-AE | 66.67 | **56.79** |

The right of fig. 9 shows the computational interface prototype that was created for demonstration. The top of the photograph shows the evolution of the reconstruction errors coming from the LSTM AE algorithm. Every second ($x$-axis), the exact errors are plotted in a graphical window (top of the photograph) and also shown in the console (bottom of the photograph). Since the demonstration was planned to be experimented by different types of persons (e.g., business managers, journalists, school students) during a two day period, two anomaly thresholds (shown as horizontal colored lines in the graphical plot) were defined: yellow – symbolizing some certainty of an anomaly, set as the maximum reconstruction error (MSE=120) that was obtained when using the validation data from the Mix6 dataset (it only includes normal data); red – higher threshold that defines a stronger anomaly event and that was set empirically, prior to the pilot execution, when configuring the real in-vehicle AAD system setup. It should be noted that the BCM in-vehicle audio contains a higher level of ambient noise, thus the pilot reconstruction errors are much higher when compared with the synthetic in-vehicle computer experiments.

During the event demonstration, the AAD system algorithm behaved as expected, tending to accurately detect a cough. In order to measure the results,

**Table 7** Accuracy obtained by the AAD system during the demonstration event

| Cough location | Air Conditioner | Radio | Windows | Engine | %Accuracy |
|---|---|---|---|---|---|
| Driver seat | | | | | 100 |
| Front passenger | On | Off | Closed | On | 100 |
| Back Seats | | | | | 90 |
| Driver seat | | | | | 100 |
| Front passenger | On | On | Closed | On | 100 |
| Back Seats | | | | | 80 |
| Driver seat | | | | | 100 |
| Passanger | On | On | Open | On | 100 |
| Back Seats | | | | | 80 |

3 scenarios were created, all assuming a non moving vehicle: 1 – radio off and windows closed; 2 – radio on and windows closed; and 3 – radio on and windows open. For each scenario, 3 person positions were assumed: driver seat, front passenger seat and back seats. For each position, a total of 10 coughs were generated. The obtained average AAD system classification accuracy results, when considering the first threshold (yellow colored in he right of fig. 9), are presented in table 7. Overall, a high quality AAD performance was achieved, ranging from 80% (third scenario, back seats) to 100% (all front driver and passenger seat experiments).

# 4 Conclusions

In this paper, three AutoEncoder (AE) deep learning architectures are proposed for an unsupervised Acoustic Anomaly Detection (AAD) task: a Dense AE, a Convolutional Neural Network (CNN) AE and a Long Short-Term Memory Autoencoder (LSTM) AE). The three AE architectures were first applied to six different real-world industrial machine sound datasets. Using development records from the datasets and sound energy features from mel-spectrograms to preprocess the raw sounds, several preliminary experiments were conducted in order to tune the AE hyperparameters, namely in terms of hidden layers and nodes and activation functions. Then, the selected AE architectures were trained and tested using the evaluation instances from the public domain datasets. Overall, competitive results were obtained by the proposedAEs when compared with a recently baseline AE architecture [26]. Then, on a second experimentation stage, the previously tuned AE architectures were adapted to model audio from a different application domain of in-vehicle intelligence, which was the main target of the joint research with *Bosch Car Multimedia S.A.* (BCM). In particular, three in-vehicle usage scenarios were explored by assuming a synthetic and realistic in-vehicle sound simulator [6]. In both application domains (working machines and in-vehicle), a high quality AAD discrimination was obtained, ranging from 72% to 91% (working machine data) and from 78% to 81% (in-vehicle audio). In particular, the best in-vehicle results were obtained by the CNN and LSTM AEs. In collaboration with the BCM company, an AAD intelligent system prototype was developed, assuming a selected LSTM AE model (trained with one of the synthetic sound

mixtures) for a real in-vehicle pilot demonstration that involved the cough use case. A very positive feedback was obtained, with the AAD intelligent system presenting a high cough detection accuracy (e.g., 100% for the front seat passengers).

As future work, different deep learning architectures for AAD will be explored, such as Variational AEs [46]. Furthermore, the effect of using audio data augmentation techniques (e.g., pitching, time-shifting, Generative Adversarial Networks) or signal frequency filtering tools will also be studied, aiming to further improve the AAD results.

## Acknowledgments

This work is supported by the European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) - Project n 039334; Funding Reference: POCI-01-0247-FEDER-039334.

## Compliance with ethical standards

**Conflict of interest** – The authors declare no conflict of interest.

## References

[1] Zhu, T., Wang, J., Cheng, S., Li, Y., Li, J.: Retrieving the relative kernel dataset from big sensory data for continuous queries in IoT systems. Eurasip Journal on Wireless Communications and Networking **2019**(1) (2019). https://doi.org/10.1186/s13638-019-1467-4

[2] Koizumi, Y., Saito, S., Uematsu, H., Kawachi, Y., Harada, N.: Unsupervised detection of anomalous sound based on deep learning and the neyman–pearson lemma. IEEE/ACM Transactions on Audio, Speech, and Language Processing **27**(1), 212–224 (2018)

[3] Panfilenko, D., Poller, P., Sonntag, D., Zillner, S., Schneider, M.: Bpmn for knowledge acquisition and anomaly handling in cps for smart factories. In: 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1–4 (2016). IEEE

[4] Sonntag, D., Zillner, S., van der Smagt, P., Lörincz, A.: Overview of the cps for smart factories project: Deep learning, knowledge acquisition, anomaly detection and intelligent user interfaces. In: Industrial Internet of Things, pp. 487–504. Springer, Cham (2017)

[5] Kim, S., Chang, J.J.E., Park, H.H., Song, S.U., Cha, C.B., Kim, J.W., Kang, N.: Autonomous taxi service design and user experience. International Journal of Human-Computer Interaction **36**(5), 429–448 (2020)

[6] Pereira, P.J., Coelho, G., Ribeiro, A., Matos, L.M., Nunes, E.C., Ferreira, A.L., Pilastri, A.L., Cortez, P.: Using deep autoencoders for in-vehicle audio anomaly detection. In: Watróbski, J., Salabun, W., Toro, C., Zanni-Merk, C., Howlett, R.J., Jain, L.C. (eds.) Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 25th International Conference KES-2021, Virtual Event / Szczecin, Poland, 8-10 September 2021. Procedia Computer Science, vol. 192, pp. 298–307 (2021)

[7] Duman, T.B., Bayram, B., İnce, G.: Acoustic anomaly detection using convolutional autoencoders in industrial processes. In: International Workshop on Soft Computing Models in Industrial and Environmental Applications, pp. 432–442 (2019). Springer

[8] Aurino, F., Folla, M., Gargiulo, F., Moscato, V., Picariello, A., Sansone, C.: One-class svm based approach for detecting anomalous audio events. In: 2014 International Conference on Intelligent Networking and Collaborative Systems, pp. 145–151 (2014). IEEE

[9] Wang, X., Jin, B., Du, Y., Cui, P., Tan, Y., Yang, Y.: One-class graph neural networks for anomaly detection in attributed networks. Neural Computing and Applications **33**(18), 12073–12085 (2021). https://doi.org/10.1007/s00521-021-05924-9

[10] Mishra, P.K., Gautam, C., Tiwari, A.: Minimum variance embedded auto-associative kernel extreme learning machine for one-class classification. Neural Computing and Applications **33**(19), 12973–12987 (2021)

[11] Harar, P., Galaz, Z., Alonso-Hernandez, J.B., Mekyska, J., Burget, R., Smekal, Z.: Towards robust voice pathology detection. Neural Computing and Applications, 1–11 (2018)

[12] Farzad, A., Gulliver, T.A.: Unsupervised log message anomaly detection. ICT Express **6**(3), 229–237 (2020)

[13] Rovetta, S., Mnasri, Z., Masulli, F.: Detection of hazardous road events from audio streams: An ensemble outlier detection approach. In: 2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS), pp. 1–6 (2020). IEEE

[14] Kohlsdorf, D., Herzing, D., Starner, T.: An auto encoder for audio dolphin communication. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–7 (2020). IEEE

[15] Oh, D.Y., Yun, I.D.: Residual error based anomaly detection using auto-encoder in smd machine sound. Sensors **18**(5), 1308 (2018)

[16] Koizumi, Y., Saito, S., Yamaguchi, M., Murata, S., Harada, N.: Batch Uniformization for Minimizing Maximum Anomaly Score of DNN-based Anomaly Detection in Sounds (2019)

[17] Ribeiro, D., Matos, L.M., Cortez, P., Moreira, G., Pilastri, A.L.: A Comparison of Anomaly Detection Methods for Industrial Screw Tightening. In: Gervasi, O., et al. (eds.) Computational Science and Its Applications - ICCSA 2021 - 21st International Conference, Cagliari, Italy, September 13-16, 2021, Proceedings, Part II. LNCS, vol. 12950, pp. 485–500 (2021)

[18] Coelho, G., Pereira, P., Matos, L.M., Ribeiro, A., Nunes, E.C., Ferreira, A.L., Cortez, P., Pilastri, A.L.: Deep dense and convolutional autoencoders for machine acoustic anomaly detection. In: Maglogiannis, I., MacIntyre, J., Iliadis, L. (eds.) Artificial Intelligence Applications and Innovations - 17th IFIP WG 12.5 International Conference, AIAI 2021, Hersonissos, Crete, Greece, June 25-27, 2021, Proceedings. IFIP Advances in Information and Communication Technology, vol. 627, pp. 337–348 (2021)

[19] Koizumi, Y., Saito, S., Uematsu, H., Harada, N., Imoto, K.: Toyadmos: A dataset of miniature-machine operating sounds for anomalous sound detection. In: 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), pp. 313–317 (2019). IEEE. https://ieeexplore.ieee.org/document/8937164

[20] Purohit, H., Tanabe, R., Ichige, T., Endo, T., Nikaido, Y., Suefusa, K., Kawaguchi, Y.: MIMII Dataset: Sound dataset for malfunctioning industrial machine investigation and inspection. In: Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019), pp. 209–213 (2019)

[21] Provotar, O.I., Linder, Y.M., Veres, M.M.: Unsupervised anomaly detection in time series using lstm-based autoencoders. In: 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT), pp. 513–517 (2019). IEEE

[22] Tagawa, T., Tadokoro, Y., Yairi, T.: Structured denoising autoencoder for fault detection and analysis. In: Asian Conference on Machine Learning, pp. 96–111 (2015)

[23] Kawaguchi, Y., Endo, T.: How can we detect anomalies from subsampled audio signals? In: 2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1–6 (2017). IEEE

[24] Chu, S., Narayanan, S., Kuo, C.-C.J.: Environmental sound recognition with timefrequency audio features. Audio, Speech, and Language Processing, IEEE Transactions on **17**, 1142–1158 (2009). https://doi.org/10.

1109/TASL.2009.2017438

[25] Purwins, H., Li, B., Virtanen, T., Schlüter, J., Chang, S., Sainath, T.N.: Deep learning for audio signal processing. IEEE J. Sel. Top. Signal Process. **13**(2), 206–219 (2019). https://doi.org/10.1109/JSTSP.2019.2908700

[26] Koizumi, Y., Kawaguchi, Y., Imoto, K., Nakamura, T., Nikaido, Y., Tanabe, R., Purohit, H., Suefusa, K., Endo, T., Yasuda, M., Harada, N.: Description and discussion on DCASE2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring. CoRR **abs/2006.05822** (2020)

[27] Smith, S.W.: The Scientist and Engineer's Guide to Digital Signal Processing Chapter 25 Special Imaging Techniques, p. 28 (1999)

[28] Jam, M.M., Sadjedi, H.: Identification of hearing disorder by multi-band entropy cepstrum extraction from infant's cry. In: 2009 International Conference on Biomedical and Pharmaceutical Engineering, pp. 1–5 (2009)

[29] Afrillia, Y., Mawengkang, H., Ramli, M., Fadlisyah, Fhonna, R.P.: Performance measurement OfMel frequency ceptral coefficient(MFCC) method in learning system of al- qur'an based InNaghamPattern recognition. Journal of Physics: Conference Series **930**, 012036 (2017). https://doi.org/10.1088/1742-6596/930/1/012036

[30] Bradbury, J., Frostig, R., Hawkins, P., Johnson, M.J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., Zhang, Q.: JAX: composable transformations of Python+NumPy programs (2018). http://github.com/google/jax

[31] Liu, Y., Zhuang, C., Lu, F.: Unsupervised Two-Stage Anomaly Detection (2021)

[32] Charte, D., Charte, F., Garca, S., del Jesus, M.J., Herrera, F.: A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines. Information Fusion **44**, 78–96 (2018). https://doi.org/10.1016/j.inffus.2017.12.007

[33] An, J., Cho, S.: Variational autoencoder based anomaly detection using reconstruction probability. Special Lecture on IE **2**(1), 1–18 (2015)

[34] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448–456 (2015). PMLR

[35] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25**, 1097–1105 (2012)

[36] Li, J., Dai, W., Metze, F., Qu, S., Das, S.: A comparison of deep learning methods for environmental sound detection. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 126–130 (2017). IEEE

[37] Hershey, S., Chaudhuri, S., Ellis, D.P., Gemmeke, J.F., Jansen, A., Moore, R.C., Plakal, M., Platt, D., Saurous, R.A., Seybold, B., *et al.*: Cnn architectures for large-scale audio classification. In: 2017 Ieee International Conference on Acoustics, Speech and Signal Processing (icassp), pp. 131–135 (2017). IEEE

[38] Chen, C., Yuan, W., Xie, Y., Qu, Y., Tao, Y., Song, H., Ma, L.: Novelty detection via non-adversarial generative network. arXiv preprint arXiv:2002.00522 (2020)

[39] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT press, Cambridge, Massachusetts (2016). http://www.deeplearningbook.org

[40] Nguyen, H., Tran, K.P., Thomassey, S., Hamad, M.: Forecasting and anomaly detection approaches using lstm and lstm autoencoder techniques with the applications in supply chain management. International Journal of Information Management **57**, 102282 (2021)

[41] Clevert, D., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). In: ICLR (Poster) (2016)

[42] Brownlee, J.: Long Short-term Memory Networks with Python: Develop Sequence Prediction Models with Deep Learning, (2017)

[43] Gonçalves, S., Cortez, P., Moro, S.: A deep learning classifier for sentence classification in biomedical and computer science abstracts. Neural Computing and Applications **32**(11), 6793–6807 (2020)

[44] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems (2016)

[45] Hand, D.J.: Assessing the performance of classification methods. International Statistical Review **80**(3), 400–414 (2012)

[46] Ntalampiras, S., Potamitis, I.: Acoustic detection of unknown bird species and individuals. CAAI Transactions on Intelligence Technology (2021). https://doi.org/10.1049/cit2.12007