DEEP DENSE AND CONVOLUTIONAL AUTOENCODERS FOR UNSUPERVISED ANOMALY DETECTION IN MACHINE CONDITION SOUNDS

Technical Report

Alexandrine Ribeiro[1]*, Luís Miguel Matos*[2]*, Pedro José Pereira*[2]*, Eduardo C. Nunes*[2]*, André L. Ferreira*[3]*, Paulo Cortez*[2]*, André Pilastri*[1]

[1] EPMQ - IT Engineering Maturity and Quality Lab, CCG ZGDV Institute, Guimarães, Portugal, {alexandrine.ribeiro, andre.pilastri}@ccg.pt
2 ALGORITMI Centre, Dep. Information Systems, University of Minho, Guimarães, Portugal, {luis.matos, pedro.pereira, pcortez}@dsi.uminho.pt
{b12176}@algoritmi.uminho.pt
[3] Bosch Car Multimedia, Portugal,{Andre.Ferreira2}@pt.bosch.com

ABSTRACT

This technical report describes two methods that were developed for Task 2 of the DCASE 2020 challenge. The challenge involves an unsupervised learning to detect anomalous sounds, thus only normal machine working condition samples are available during the training process. The two methods involve deep autoencoders, based on dense and convolutional architectures that use mel-spectogram processed sound features. Experiments were held, using the six machine type datasets of the challenge. Overall, competitive results were achieved by the proposed dense and convolutional AE, outperforming the baseline challenge method.

Index Terms— DCASE 2020 Challenge, Autoencoder, Convolutional neural network.

1. INTRODUCTION

This work is motivated by a real-world task from the challenge on Detection and Classification of Acoustic Scenes and Events (DCASE): unsupervised Anomalous Sound Detection (ASD). The DCASE challenge had its first edition in 2013 and three more editions from 2016 to 2019, with distinct learning tasks, ranging from acoustic scene classification to sound event detection.

In this technical report, we address the second task from the current DCASE edition (2020): Unsupervised Detection of Anomalous Sounds for Machine Condition Monitoring [1]. The task aims to automatically detect, and as soon as possible, if a given machine is not working correctly by using only on the sound produced by the machine. Such an anomaly detection model is thus value for preventing future machine issues (e.g., equipment damage). The main challenge is to detect abnormal sounds using only standard working machine sound samples, assuming that the sounds produced by mechanical anomalies on the equipment are unknown. Although it may seem a binary classification problem ("normal" or "anomaly"), since the models can only be trained using data from one class, this task must be solved with an unsupervised learning anomaly detection approach.

For this task, a baseline system implementation was provided for comparison purposes [1]. The baseline consists of a dense Autoencoder (AE) with three layers, in both the encoder and decoder components, with 128 units, and a latent space with 8 units, all with the ReLU activation function. In this paper, we propose two deep learning models, based on a Dense and Convolutional architectures fed with mel-spectograms, which are further detailed in the next section.

2. METHODS

2.1. Datasets

The data used for this task comprises parts of ToyADMOS [2] and the MIMII Dataset [3] consisting of the normal and anomalous operating sounds of six types of toy/real machines. This data was provided in two datasets (development and evaluation) for 6 different machine types: ToyCar, ToyConveyor, slider, pump, fan, and valve. In the development dataset, each machine type has 4 different machines, except for ToyConveyor, which has only 3. Moreover, normal and anomaly labels were provided for the test data, such that the anomaly detection performance could be estimated and the model could be tuned accordingly. Regarding the evaluation data, it contains data for new machines in each machine type, both for model training and testing. Moreover, no labels are provided. A different number of approximately 10 second Waveform Audio File (WAV) files is provided for each machine. Table 1 summarizes the challenge datasets.

2.2. Autoencoders

The base learner is based on a AE, which has obtained good results in several studies [4, 5, 6, 7]. The AE is a specific artificial neural network in which the input is expected to be equal to the output and there are several hidden layers with fewer nodes than the number of inputs. The AE learning goal is to produce the same output for the same input, thus encoding and decoding the input signal via the hidden processing layers.

The encoder component of the AE maps the input vector (the features) into an hidden representation with a lower dimensional space, via a nonlinear transform. Then, the decoder component attempts to reconstruct the reverse transform, from the hidden representation to the original input signal. The difference between the

Table 1: Summary of provided datasets

Machine Type	Mode	Machine ID	Audio Files	
			Train	Test
ToyCar	Dev.	01	1000	614
		02	1000	615
		03	1000	615
		04	1000	615
	Eval.	05	1000	515
		06	1000	515
		07	1000	515
ToyConveyor	Dev.	01	1000	1200
		02	1000	1155
		03	1000	1154
	Eval.	04	1000	555
		05	1000	555
		06	1000	555
fan	Dev.	00	911	507
		02	916	549
		04	933	448
		06	915	461
	Eval.	01	934	426
		03	912	458
		05	1000	458
pump	Dev.	00	906	243
		02	905	211
		04	602	200
		06	936	202
	Eval.	01	903	216
		03	606	213
		05	908	348
slider	Dev.	00	968	456
		02	968	367
		04	434	278
		06	434	189
	Eval.	01	968	278
		03	968	278
		05	434	278
valve	Dev.	00	891	219
		02	608	220
		04	900	220
		06	892	220
	Eval.	01	679	220
		03	863	220
		05	899	500

original input vector and the AE output response is called the reconstruction error [8].

In this challenge, the reconstruction error element is used to detect sound anomalies. Firstly, an AE is trained with only normal sound samples, aiming to minimize the reconstruction error. The obtained model is assumed to be capable of compressing the input features, learning their most relevant relationships. Secondly, the trained AE can be tested with unseen data. If the unseen data is similar to the trained patterns (related to the normal sounds), when the AE should reproduce the new input with good accuracy. However, if the unseen data is anomalous, the AE should not be able to reconstruct the input and the error will be greater. Thus, the magnitude of the reconstruction error can be used to detect anomalies. The proposed unsupervised anomaly detection approaches consist of simple AE networks. The systems were modeled to be generic, only changing the training data fed to the model, hence creating a generic model for anomaly detection in several machines.

2.3. Dense Autoencoder

The first approach proposed, consists of a deep fully-connected AE (top of Figure 1). After performing several preliminary experiments with different architectures (varying number of layers, hidden units, activation function and latent space dimensions), a final dense AE architecture was selected. The encoder and decoder networks consist of four fully-connected layers with 512 hidden units, followed by Batch Normalization and ReLU as the activation function. The bottleneck layer is set as one fully-connected layer with 8 hidden units, resulting in a 8-dimensional latent space.

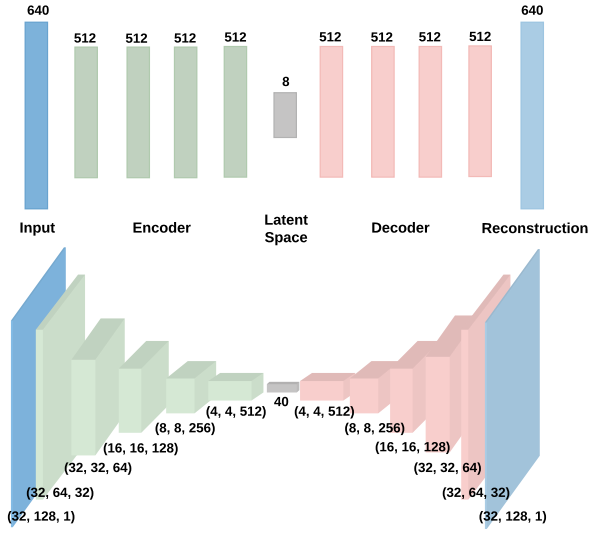

Figure 1: Proposed AE Network architectures: Dense AE (top) and Convolutional AE (bottom).

2.4. Convolutional Autoencoder

Recently, Convolutional Neural Networks (CNNs) have been increasingly applied for audio processing tasks by using audio spectrograms as features [9, 10, 11]. CNNs are an effective way to capture spatial information from multidimensional data being naturally suitable for exploring image-like time-frequency representations of audio, such as spectrograms. The main goal of a CNN is to learn local structure in input data. Locality is a key property of CNNs. This is accomplished by convolutional filters that are applied to local regions of the previous layers to capture local patterns. Consequently, spatial features must be locally correlated. As time-frequency representations of audio are treated as images by CNN architectures, these features should be locally correlated in the sense of time and frequency.

The second approach proposed for the ASD task consists of a deep CNN AE (shown in the bottom of Figure 1). Similarly to the dense AE network, preliminary experiments were used to adjust

the CNN AE. The encoder and decoder networks are comprised of convolutional layers with Batch Normalization and the ReLU activation function after each convolution. The encoder network consists in a stack of five hidden layers with convolutional filters of 32, 64, 128, 256, and 512, kernel sizes of 5, 5, 5, 3, and 3, and strides of $(1, 2)$, $(1, 2)$, $(2, 2)$, $(2, 2)$, and $(2, 2)$, respectively. The bottleneck consists of a convolutional layer with 40 convolutional filters, reducing the encoder feature maps to a 40-dimensional compressed representation of the input. Regarding the decoder network, first a fully-connected layer inflates the latent space to the shape the last layer of the encoder, followed by five transposed convolutional layers that mirror of the encoder layers.

2.5. Audio Features

Regarding the feature engineering process, we have initially considered two main sound processing methods: Mel Frequency Cepstral Coefficients (MFCCs) and Mel Frequency Energy Coefficients (MFECs). MFCCs, which are derived from the mel-cepstrum representation of the audio, are one of the best knowns and most popular audio processing features [12]. However, when computing MFCCs, a Discrete Cosine Transform (DCT) is applied to the logarithm of the filter bank outputs, resulting in decorrelated MFCC features. Therefore, they have the drawback of having non-local features, which makes them unsuitable for CNN processing. As such, in this work we explored a different feature for audio signal processing named MFECs, which are log-energies derived directly from the filter-banks energies. These are similar to MFCCs, however, they do not include the DCT operation. This feature provided good results in detecting different audio sounds and classification of sounds [13, 14]. Therefore, MFECs were selected as audio features for the proposed systems.

3. EXPERIMENTS AND RESULTS

In this section, we describe our pipeline, including the feature preprocessing, model settings, hyperparameters and results obtained for both AE architectures.

3.1. Features Extraction

In the Dense Autoencoder system, audio data is buffered in fixed-length 1 second intervals with a 50% overlap. For each audio buffer obtained, the segment is then divided into 64 ms analysis frames, with a 50% overlap and 128 MFECs extracted from the magnitude spectrum of each frame. Then, a context window of size 5 is used. Thus, 5 frames are concatenated to form a 640-dimensional input vector. This representation is depicted in Figure 2.

In the Convolutional Autoencoder system, for each audio, 128 log mel-band energy features are extracted from the magnitude spectrum, considering 64 ms analysis frames with 50% overlap. Then, each feature is normalized to zero mean and unit standard deviation by using statistics from the training data. Finally, the mel spectrogram is segmented about every second into 32 column data with approximately 100 ms of hop size. This extraction procedure is shown in Figure 3.

3.2. Training Settings

The encoder and decoder were trained to minimize the Mean Squared Error (MSE) between input and its reconstruction. Both architectures were trained with a learning rate of 0.001 and the Adam optimizer. The training process is stopped early when the validation loss has stopped improving for 10 epochs and the best saved model is selected. The training procedure was iterated up to a maximum of 100 epochs. The batch size for the Dense AE and Convolutional AE algorithms was set as 512 and 64, respectively.

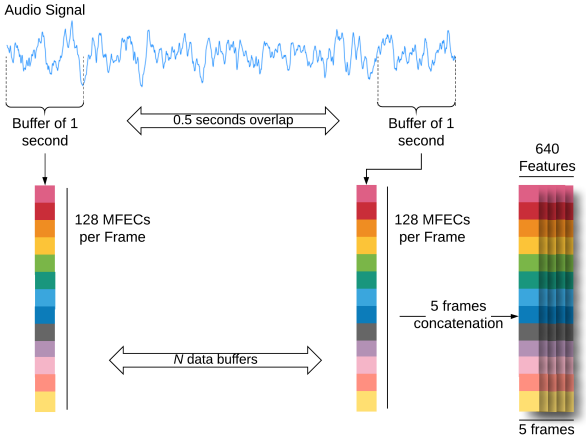

Figure 2: Feature extraction procedure of the dense AE.

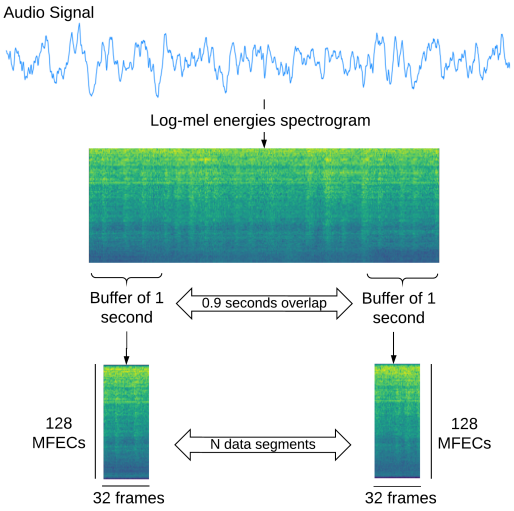

Figure 3: Feature extraction procedure of the CNN AE.

3.3. Results and Discussion

All developed architectures were implemented in Python programming language and experiments were conducted in several GPUs (Titan Xp and 1080Ti). To evaluate the model performance, both AUC and pAUC metrics were used, as defined in the task description [1]. Table 2 presents the AUC and pAUC values for each specific machine and mean values for each machine type, obtained for the development dataset by both AE models. For comparison purposes, the baseline system results are also provided in the table.

In terms of mean AUC and pAUC values for each machine type, the Dense AE outperforms the baseline system in every machine type. Furthermore, the baseline system only achieved better results in a few specific machines, namely ToyCar ID 3, pump IDs

Table 2: Performance results for DCASE 2020 Task 2 for the development dataset (best mean values in **bold**)

Machine Type	Machine ID	Baseline		Dense AE		Conv AE	
		AUC (%)	pAUC (%)	AUC (%)	pAUC (%)	AUC (%)	pAUC (%)
ToyCar	1	81.36	68.40	83.87	72.64	81.59	71.88
	2	85.97	77.72	87.56	80.35	85.46	79.92
	3	63.30	55.21	63.12	55.02	62.73	55.08
	4	84.45	68.97	88.60	76.68	82.38	69.60
	Average	78.77	67.58	**80.79**	**71.17**	78.04	69.12
ToyConveyor	1	78.07	64.25	81.67	69.41	79.90	62.71
	2	64.16	56.01	68.04	58.31	67.78	54.85
	3	75.35	61.03	79.59	63.64	80.11	62.53
	Average	72.53	60.43	**76.43**	**63.79**	75.93	60.03
fan	0	54.41	49.37	56.73	49.72	51.77	49.05
	2	73.40	54.81	79.60	54.00	72.71	55.51
	4	61.61	53.26	70.11	54.11	62.60	52.80
	6	73.92	52.35	81.69	55.15	80.05	53.19
	Average	65.83	52.45	**72.03**	**53.25**	66.78	52.63
pump	0	67.15	56.74	66.94	56.83	66.37	54.95
	2	61.53	58.10	60.77	60.31	54.31	53.58
	4	88.33	67.10	87.00	66.32	94.64	77.26
	6	74.55	58.02	77.53	60.32	76.97	58.05
	Average	72.89	59.99	**73.06**	60.94	72.07	**60.96**
slider	0	96.19	81.44	96.12	82.30	98.86	94.47
	2	78.97	63.68	79.55	64.42	84.06	69.33
	4	94.30	71.98	95.44	76.14	97.69	87.82
	6	69.59	49.02	77.22	49.56	86.46	53.16
	Average	84.76	66.53	87.08	68.10	**91.77**	**76.20**
valve	0	68.76	51.70	74.61	52.28	78.69	52.59
	2	68.18	51.83	76.68	52.72	85.02	55.92
	4	74.30	51.97	79.58	50.96	82.59	53.68
	6	53.90	48.43	57.78	48.73	69.03	50.22
	Average	66.28	50.98	72.16	51.17	**78.83**	**53.10**

0, 2 and 4, and slider ID 0. Regarding the CNN AE, in general the model outperformed the baseline system, although the latter achieved higher mean AUC values for 2 of 6 machine types (ToyCar and pump). The two proposed AE are quite competitive in terms of mean AUC and pAUC values, with CNN AE outperforming Dense AE only in 2 of 6 machine types (slider and valve). Overall, both the dense and CNN AE outperform the baseline system in both anomaly detection metrics (AUC and pAUC). Considering that none of the proposed AE models obtained the best results for all machine types, we have created a third method for the competition, which is termed mixed approach. This third approach uses the best AE for each machine type, namely the CNN AE is used for the slider and valve machines, while the Dense AE is adopted for the other machine types. All the developed code is available on github[1].

4. CONCLUSIONS

In this paper, we proposed two Autoencoder (AE) models for an unsupervised Anomalous Sound Detection (ASD), for the second task of the DCASE 2020 challenge. The AE models are based on Dense and Convolutional Neural Networks (CNN). Several preliminary experiments were conducted, resulting in two proposed AE architectures that use sound energy features from mel-spectograms. Using the provided challenge datasets, the two deep AE were trained and tested with the competition six machine types. Overall, competitive results were obtained when compared with the challenge baseline model. For two machine types (slider and valve), the best results were achieved by the CNN AE, while the Dense AE provided the best results for the other machines (ToyCar, ToyConveyor, fan and pump). Thus, a third method was proposed for the competition, which uses the best AE model for each machine type. We consider that the obtained AE results are of quality. For instance, the achieved test data AUC values range from 72% (good) to 92% (excellent discrimination).

As future work, we aim to explore with more depth the proposed AE structures. For instance, by adopting audio data augmentation techniques (e.g., pitching, time-shifting) to improve the training results. Furthermore, we intend to explore other neural network architectures for sound anomaly detection, such as Generative Adversarial Networks (GAN) and Variational AEs.

5. 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^{o} 039334; Funding Reference: POCI-01-0247-FEDER-039334.

[1] https://github.com/APILASTRI/DCASE_Task2_UMINHO

6. REFERENCES

[1] Y. Koizumi, Y. Kawaguchi, K. Imoto, T. Nakamura, Y. Nikaido, R. Tanabe, H. Purohit, K. Suefusa, T. Endo, M. Yasuda, and N. Harada, "Description and discussion on dcase2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring," 2020.

[2] Y. Koizumi, S. Saito, H. Uematsu, N. Harada, and K. Imoto, "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)*. IEEE, 2019, pp. 313–317. [Online]. Available: https://ieeexplore.ieee.org/document/8937164

[3] H. Purohit, R. Tanabe, T. Ichige, T. Endo, Y. Nikaido, K. Suefusa, and Y. Kawaguchi, "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)*, November 2019, pp. 209–213. [Online]. Available: http://dcase.community/documents/workshop2019/proceedings/DCASE2019Workshop_Purohit_21.pdf

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

[5] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised detection of anomalous sound based on deep learning and the neyman–pearson lemma," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 212–224, 2018.

[6] T. Tagawa, Y. Tadokoro, and T. Yairi, "Structured denoising autoencoder for fault detection and analysis," in *Asian Conference on Machine Learning*, 2015, pp. 96–111.

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

[8] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lecture on IE*, vol. 2, no. 1, pp. 1–18, 2015.

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

[10] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017, pp. 131–135.

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

[12] G. Sharma, K. Umapathy, and S. Krishnan, "Trends in audio signal feature extraction methods," *Applied Acoustics*, vol. 158, p. 107020, 2020.

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

[14] A. Torfi, S. M. Iranmanesh, N. M. Nasrabadi, and J. M. Dawson, "3d convolutional neural networks for cross audio-visual matching recognition," *IEEE Access*, vol. 5, pp. 22 081–22 091, 2017. [Online]. Available: https://doi.org/10.1109/ACCESS.2017.2761539