FKIE-VUB SYSTEM FOR DCASE2024 CHALLENGE TASK 2: FIRST-SHOT UNSUPERVISED ANOMALOUS SOUND DETECTION FOR MACHINE CONDITION MONITORING

Technical Report

Kevin Wilkinghoff

Fraunhofer FKIE
Fraunhoferstraße 20, 53343 Wachtberg, Germany
kevin.wilkinghoff@ieee.org

Yacine Bel-Hadj

Vrije Universiteit Brussel (VUB), OWI-lab
Pleinlaan 2, 1050 Brussels, Belgium
yacine.bel-hadj@vub.be

ABSTRACT

This report contains a description of the FKIE-VUB system submitted to task 2 “First-Shot Unsupervised Anomalous Sound Detection for Machine Condition Monitoring” of the DCASE2024 Challenge. The submitted system is an adaptation of a previously proposed system based on an embedding model trained with an auxiliary classification task, which is imposed by self-supervised learning and provided meta information. The main modifications of the presented system are to replace the sub-cluster AdaCos loss with the AdaProj loss and to use balanced sample weights when training the embedding model. In experimental evaluations, it is shown that both modifications improve the resulting performance and that the proposed system significantly outperforms both baseline systems of the challenge as well as the model it is based on.

Index Terms— anomalous sound detection, machine listening, anomaly detection, representation learning, domain generalization

1. INTRODUCTION

Task 2 of the DCASE2024 Challenge [1] is called “First-Shot Unsupervised Anomalous Sound Detection for Machine Condition Monitoring”. This year, submitted systems not only had to be trained using only normal data and to be robust to possible acoustic domain shifts, which may for example be caused by changing machine parameters or the background noise, but also had to be capable to effectively handle completely novel machine types without having access to recordings of similar machines and not always having access to machine parameter settings. Such constraints are vital for practical industrial applications where data annotations may not always be available and domain shifts frequently occur. Needing to re-design the system for different machine types, re-train the entire system when changing machine settings or to always label extensive amounts of data for every machine type is costly and highly impractical.

The organizers provided two baseline systems based on machine-type specific autoencoders with different ways to calculate anomaly scores: 1) using the mean squared error (MSE) and 2) using the Mahalanobis distance (MAHALA) [2]. The dataset of the challenge consists of a development set and an evaluation set with mutually exclusive machine types, which both contain a training split consisting of only normal data and a test split consisting of normal and anomalous data from ToyAdmos2 [3] and MIMII-DG [4]. Each training split contains 990 samples belonging to the source domain and 10 samples belonging to the target domain. The test splits contain about the same number of samples for both domains. For approximately 50% of the machine types additional meta information denoting machine parameter settings or models or noise conditions are available for the recordings contained in the training split. For each recording, it is known which machine type is contained in the recording. More details about the design and rules of the task as well as the dataset can be found in [1].

State-of-the-art anomalous sound detection (ASD) systems for acoustic machine condition monitoring usually utilize an embedding model to project the data into a relatively low-dimensional vector space and measure distances to or estimate distributions of normal data to determine whether a test sample is normal or anomalous. The main difficulty of designing such an ASD system is choosing how to train the embedding model. Here, many of the best-performing systems utilize auxiliary classification tasks based on classes defined by provided meta information [5] or self-supervised learning (SSL) [6]. These models have been shown to outperform one-class models such as autoencoders because utilizing meta information enables the system to closely monitor the sounds of the target machine and ignore the background noise [7].

The main contribution of this work is to present a conceptually simple state-of-the-art ASD system[1] for the DCASE2024 Challenge. Apart from the difficulties imposed by the rules of the challenge task and the design of the dataset, we included the following difficulties to simplify the system itself and make it more useful for practical applications:

- only a single embedding model is used for all machine types and domains
- no samples belonging to the target domain are used for training the embedding model to really capture the difficulty of domain generalization
- no external datasets are used to train the system

In experimental evaluations conducted on the development set of the DCASE2024 ASDdataset, it will be shown that the proposed system significantly outperforms both baseline systems of the challenge.

2. SYSTEM DESCRIPTION

The system described in this report is largely based on the ASD system presented in [6]. As this system yields state-of-the-art perfor-

[1]An open-source implementation of the proposed system is available at: `https://github.com/wilkinghoff/dcase2024_task2`

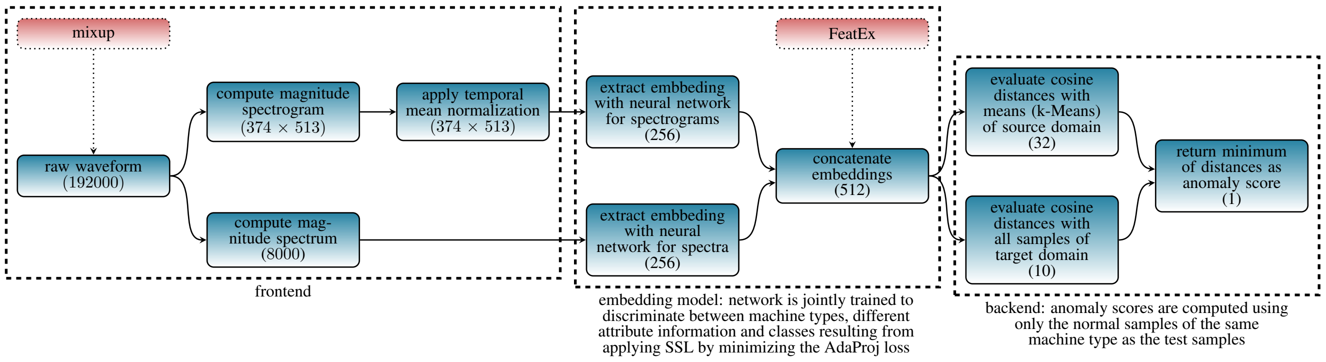
mixup
raw waveform (192000)
compute magnitude spectrogram (374 × 513)
apply temporal mean normalization (374 × 513)
compute magnitude spectrum (8000)
frontend
FeatEx
extract embedding with neural network for spectrograms (256)
extract embedding with neural network for spectra (256)
concatenate embeddings (512)
embedding model: network is jointly trained to discriminate between machine types, different attribute information and classes resulting from applying SSL by minimizing the AdaProj loss
evaluate cosine distances with means (k-Means) of source domain (32)
evaluate cosine distances with all samples of target domain (10)
return minimum of distances as anomaly score (1)
backend: anomaly scores are computed using only the normal samples of the same machine type as the test samples

Figure 1: Overall architecture of the ASD system. Representation size in each step is given in brackets. This figure is adapted from [6] and originally adapted from [8].

mance on the DCASE2023 ASD dataset [9], it is a viable choice as a starting point and will also serve as an additional baseline system in the experimental evaluations of this report. The overall architecture of the proposed system can be found in Figure 1 and can be divided into the three main processing blocks 1) a frontend, 2) an embedding model and 3) a backend. In the following, these three blocks will be described in detail with an emphasis on all modifications of the ASD system it is based on [6].

2.1. Frontend

The frontend consists of pre-processing the data and extracting two different feature representations for the embedding model. First of all, waveforms that are shorter than $192,000$ samples, i.e. shorter than $12\,\mathrm{s}$ are padded with zeros before and after the original signal content. This is the longest duration of any of the recordings contained in the dataset. Then, two different input features are extracted. As a first input feature, a magnitude spectrogram with a window size of 1024 and a hop size of 512 is used. For each magnitude spectrogram, the temporal mean, calculated by using only the frames belonging to the non-padded signal values, is subtracted to remove constant frequency information. As a second input feature, the magnitude of the spectrum belonging to the entire waveform is used. In contrast to the system presented in [6], only the frequencies up to $8\,\mathrm{kHz}$, which is equal to half of the sampling rate, are kept. To prevent overfitting of the embedding model, mixup [10] with a mixing coefficient drawn from a uniform distribution is randomly applied with a probability of 0.5 to the waveforms.

2.2. Embedding model

The embedding model consists of two different convolutional neural networks (CNNs) as sub-networks, one for each of the two feature branches, and has the same general architecture as the embedding model used in [6]. The sub-network for the spectrogram branch has a modified ResNet architecture [11] with four residual blocks, a max-pooling operation over the time dimension in combination with a flattening operation and a linear layer. The sub-network for the spectrum branch consists of three one-dimensional convolutions with large strides are applied to downsample the input followed by a flattening operation and and five dense layers. In both networks, ReLU is chosen as an activation function and batch normalization [12] is applied. To obtain a single embedding for each input sample, the embedding of both sub-networks are concatenated. More details about the network architecture can be found in [8].

2.3. Training strategy

To train the neural network two equally weighted loss terms capturing two different auxiliary classification tasks are used as proposed in [6], i.e. for sample x and class label y

$$\mathcal{L}_{\text{total}}(x,y) = \mathcal{L}_{\text{meta}}(x,y) + \mathcal{L}_{\text{ssl}}(x_{\text{ssl}}, y_{\text{ssl}})$$

with x_{ssl} and y_{ssl} being defined by an SSL approach as described below and $\mathcal{L}_{\text{meta}}$ and $\mathcal{L}_{\text{ssl}}$ denoting categorical crossentropies. For the first task $\mathcal{L}_{\text{meta}}$, the classes are defined by all possible combinations of machine types and specific values for provided attribute information. This is a commonly used definition of an auxiliary classification task for ASD. As an additional SSL task $\mathcal{L}_{\text{ssl}}$, feature exchange (FeatEx) is applied with a probability of 50%. Here, the embeddings of both sub-networks are randomly exchanged between two different samples of a batch and the network needs to predict whether the two embeddings belong to the same sample or not. This is similar to the training objective of the look, listen and learn embeddings [13, 14, 15] where there are two sub-networks for an audio clip and a video frame. When applying FeatEx, the network also needs to predict the exact classes the two embeddings of the sub-networks belong to. More details about this FeatEx approach can be found in [6].

The angular margin loss AdaProj [16] with an adaptive scale parameter as defined for the AdaCos loss [17] and no margin parameter is used as a loss function for each of the two loss terms. This loss function projects the embeddings to class-specific subspaces instead to single points (centers) [18, 17] or one of multiple points [19]. This leads to better ASD performance since the solution space is much larger as an entire vector space itself, which results in learning more sophisticated distributions for each class and helps to detect anomalous samples in the joint embedding space [16]. The entire network is trained for 10 epochs using Adam [20] with a batch size of 32.

When considering different values for the different attribute information as different classes, one can think of the attribute information as introducing additional sub-classes for each machine type. This has also been explicitly captured with an hierarchical learning approach for the classes [21] but only led to very marginal gains in performance. The same is true for few-shot sound event detection (SED) [22]. Still, the different attribute values lead to highly imbalanced (sub-)classes. Even though it is only an auxiliary task not directly related to the ASD problem, it may affect the resulting ASD performance. To balance these highly imbalanced classes, sample weights were applied to the gradients during training . This was done by applying the following three steps: First, for each sample x its weight $w(x) \in \mathbb{N}_0$ was set to

$$w(x) := |\{y \in Y_{\text{class}} : \text{class}(x) \neq y\}|$$

where Y_{class} denotes the set of all classes and $\text{class}(x) \in Y_{\text{class}}$ the class of sample x. Thus, samples with attribute information for which fewer samples exist have a higher weight when adapting the training. In a second step, for each machine type the labels were normalized by setting

$$w_{\text{norm}}(x) := \frac{w(x)}{\sum_{x' \in \{x' : \text{type}(x') = \text{type}(x)\}} w(x')} \in [0, 1]$$

where $\text{type}(x)$ denotes the machine type of sample x. This ensures, that each machine type has equal weight during training and avoids creating another imbalance. Last, the sample weights are re-scaled to be able to use the same number of epochs and the same learning rate as without using the sample weights. This is achieved by dividing by the mean value of the weights.

In addition to using the AdaProj loss instead of the sub-cluster AdaCos [19] and using sample weights to balance the classes, the following two modifications of the system described in [6] are utilized when training the embedding model. First, the SSL technique statistics exchange (StatEx) [23] is not longer used because no significant differences in performance could be observed. Removing StatEx simplifies the system and reduces the number of classes and thus also the number of parameters. This is also consistent with the findings in [6] where it was shown that sometimes StatEx improved the performance but sometimes the performance slightly degraded. The second modification is to use randomly initialized and non-trainable basis vectors (formerly centers) for the SSL loss. Using non-trainable centers prevents learning trivial solutions for one-class losses [24] and has also been shown to improve the ASD performance when using angular margin losses [8]. For the other angular margin losses such as the AdaCos loss [17] or the sub-cluster AdaCos loss [19] this would not be possible because the intersection of (pairwise different) points is the empty set and thus no optimal solution for both tasks would exist. But for the AdaProj loss the intersection of two linear sub-spaces, each containing the optimal solutions for one of the subtasks, is another (lower-dimensional) subspace and in particular non-empty. Hence, when using the AdaProj loss there are still optimal solutions for both tasks when using random sub-spaces that are not adapted during training. Apart from a slightly better ASD performance in general as shown in [16], this is another advantage of using the AdaProj loss.

2.4. Backend

The goal of the backend is to calculate anomaly scores such that higher scores indicate anomalous data. Since the machine type is provided for each test sample, this is done for each machine type (and section) independently. To calculate the anomaly scores, first k-means with $k = 32$ is applied to the 990 normal training samples belonging to the source domain. Then, the minimum of all cosine distances to these means as well as all ten samples belonging to the target domain is used as an anomaly score. This backend is essentially the same as the one of the baseline system [6] and was originally proposed in [8]. The only difference is that $k = 32$ instead of $k = 16$ is used when applying k-means.

Decision thresholds for semi-supervised anomaly detection are estimated by separating the most extreme values of the anomaly scores belonging to the normal training samples from the less extreme scores and hoping that the same value is a suitable choice for separating the anomaly scores of normal and anomalous samples [25]. To this end, we assume a uniform distribution of the anomaly scores and use the 90th percentile as a decision threshold.

2.5. Submissions

Three different versions of the proposed system were submitted to the challenge to investigate the impact of designing the training dataset and the impact of the sample weights on the performance. All three systems are ensembles, each consisting of ten independently trained embedding models with the exact same architectural design as described above. The anomaly scores of the ensembled systems are combined by taking the mean of the anomaly scores. The system for submission 1 was trained using the training split of the development and the evaluation dataset while using sample weights for balancing the classes. The system for submission 2 was trained using only the training split of the development set while still using sample weights. The system for submission 3 was also trained using only the training split of the development set but without using sample weights.

3. EXPERIMENTAL RESULTS

The experimental results obtained with the three submitted systems as well as the two baseline systems of the challenge [2] and our own baseline system [6] are contained in Table 1. The following observations can be made. In terms of overall performance, our own baseline system as well as all submitted systems yield significantly better results than the two baseline systems of the challenge. This is especially true for the target domain, for which both the MSE and MAHALA baseline yield results that are close or even slightly worse than random guessing. Noticeable exceptions of this general trend are the machine types “fan” and “ToyCar”. Here, the two baseline systems of the challenge yield much better performance than the submitted systems on the source domain but very bad performance (much less than random guessing, i.e. 50% AUC) on the source domain showing that the baseline systems overfit to the source domain and do not generalize well to unseen domains.

When comparing the performance of the submitted systems to our own baseline system, it can be seen that the AUC obtained on the source domain significantly improves while the performance on the target domain is very similar. This shows that the proposed modifications not only simplify the architecture of the ASD systen but also improve the performance. Another observation to be made is that also using the additional training dataset belonging to completely different machine types than the training split of the development dataset, only improves the performance for the machine type “fan” while degrading the performance for all other machine types. Overall, the performance significantly degrades when using the additional training dataset indicating that only using training data belonging to the machine types that are evaluated seems to be a better choice. Last but not least, one can see that using sample weights for balancing the classes improves the performance when comparing the performance obtained with submission 2 to the one obtained with submission 3.

Table 1: AUCs and pAUCs per machine type obtained on the development set of the DCASE2024 ASD dataset. The last row contains the harmonic mean taken over all machine types. Highest AUCs and pAUCs in each row are highlighted in bold letters.

dataset split			baseline systems			submitted systems		
machine type	domain	metric	MSE [2]	MAHALA [2]	own baseline [6]	submission 1	submission 2	submission 3
ToyCar	source	AUC	70.1%	**74.5%**	51.5%	52.7%	60.0%	58.0%
ToyCar	target	AUC	46.9%	43.4%	47.8%	48.2%	55.4%	**55.9%**
ToyCar	mixed	pAUC	**52.5%**	49.2%	48.4%	49.4%	50.0%	51.1%
ToyTrain	source	AUC	57.9%	56.0%	65.4%	68.3%	**71.1%**	67.7%
ToyTrain	target	AUC	57.0%	42.5%	**67.8%**	53.0%	56.9%	58.9%
ToyTrain	mixed	pAUC	48.6%	48.1%	54.2%	53.5%	54.1%	**55.4%**
bearing	source	AUC	65.9%	65.2%	70.0%	70.0%	**75.6%**	75.1%
bearing	target	AUC	55.8%	55.3%	68.9%	67.2%	**73.4%**	72.0%
bearing	mixed	pAUC	50.4%	51.4%	**60.8%**	58.8%	60.3%	60.5%
fan	source	AUC	80.2%	**87.1%**	61.8%	70.3%	69.1%	70.1%
fan	target	AUC	36.2%	46.0%	60.4%	**79.9%**	65.4%	64.2%
fan	mixed	pAUC	59.0%	59.3%	58.6%	**59.5%**	54.9%	53.3%
gearbox	source	AUC	60.3%	71.9%	69.6%	70.4%	**74.2%**	68.7%
gearbox	target	AUC	60.7%	70.8%	73.8%	74.7%	**75.5%**	75.4%
gearbox	mixed	pAUC	53.2%	54.3%	57.2%	54.8%	**58.1%**	57.3%
slide rail	source	AUC	70.3%	84.0%	**94.4%**	92.3%	90.1%	90.9%
slide rail	target	AUC	48.8%	73.3%	82.6%	80.2%	**85.2%**	83.0%
slide rail	mixed	pAUC	56.4%	54.7%	67.3%	65.6%	**68.5%**	65.8%
valve	source	AUC	55.4%	56.3%	93.7%	93.4%	**95.6%**	**95.6%**
valve	target	AUC	50.7%	51.4%	**65.9%**	61.2%	55.9%	54.7%
valve	mixed	pAUC	51.2%	51.1%	65.9%	68.3%	**69.3%**	67.7%
all	source	AUC	64.8%	68.8%	69.4%	71.5%	**74.9%**	73.2%
all	target	AUC	49.6%	52.4%	65.1%	64.1%	**65.2%**	64.9%
all	mixed	pAUC	52.8%	52.4%	58.3%	57.9%	**58.6%**	58.2%

4. CONCLUSIONS

In this report, the FKIE-VUB system submitted to task 2 of the DCASE2024 Challenge was described. The system is a simplified version of a previously developed ASD system, which uses an embedding model trained with an auxiliary classification task imposed by meta information and SSL. Furthermore, two additional modifications were proposed to improve the resulting ASD performance: 1) using AdaProj as a loss function and 2) using sample weights to balance the classes during training. As the main result, the presented system significantly outperforms both baseline systems of the challenge on the DCASE2024 ASD dataset as well as the ASD system it is based on.

5. REFERENCES

[1] T. Nishida, N. Harada, D. Niizumi, D. Albertini, R. Sannino, S. Pradolini, F. Augusti, K. Imoto, K. Dohi, H. Purohit, T. Endo, and Y. Kawaguchi, "Description and discussion on dcase 2024 challenge task 2: First-shot unsupervised anomalous sound detection for machine condition monitoring," 2024, arXiv:2406.07250.

[2] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, and M. Yasuda, "First-shot anomaly detection for machine condition monitoring: A domain generalization baseline," in *31st European Signal Processing Conference (EUSIPCO)*. IEEE, 2023, pp. 191–195.

[3] N. Harada, D. Niizumi, D. Takeuchi, Y. Ohishi, M. Yasuda, and S. Saito, "ToyADMOS2: Another dataset of miniature-machine operating sounds for anomalous sound detection under domain shift conditions," in *6th Workshop on Detection and Classification of Acoustic Scenes and Events*, 2021, pp. 1–5.

[4] K. Dohi, T. Nishida, H. Purohit, R. Tanabe, T. Endo, M. Yamamoto, Y. Nikaido, and Y. Kawaguchi, "MIMII DG: sound dataset for malfunctioning industrial machine investigation and inspection for domain generalization task," in *7th Workshop on Detection and Classification of Acoustic Scenes and Events*. Tampere University, 2022, pp. 26–30.

[5] W. Junjie, W. Jiajun, C. Shengbing, S. Yong, and L. Mengyuan, "Anomalous sound detection based on self-supervised learning," DCASE2023 Challenge, Tech. Rep., June 2023.

[6] K. Wilkinghoff, "Self-supervised learning for anomalous sound detection," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 276–280.

[7] K. Wilkinghoff and F. Kurth, “Why do angular margin losses work well for semi-supervised anomalous sound detection?” *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 32, pp. 608–622, 2024.

[8] K. Wilkinghoff, “Design choices for learning embeddings from auxiliary tasks for domain generalization in anomalous sound detection,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

[9] K. Dohi, K. Imoto, N. Harada, D. Niizumi, Y. Koizumi, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi, “Description and discussion on DCASE 2023 challenge task 2: First-shot unsupervised anomalous sound detection for machine condition monitoring,” in *8th Workshop on Detection and Classification of Acoustic Scenes and Events*. Tampere University, 2023, pp. 31–35.

[10] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “Mixup: Beyond empirical risk minimization,” in *International Conference on Learning Representations (ICLR)*, 2018.

[11] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 770–778.

[12] S. Ioffe and C. Szegedy, “Batch normalization: accelerating deep network training by reducing internal covariate shift,” in *32nd International Conference on Machine Learning (ICML)*, vol. 37, 2015, pp. 448–456.

[13] R. Arandjelovic and A. Zisserman, “Look, listen and learn,” in *International Conference on Computer Vision (ICCV)*. IEEE Computer Society, 2017, pp. 609–617.

[14] ——, “Objects that sound,” in *European Conference on Computer Vision (ECCV)*, ser. Lecture Notes in Computer Science, vol. 11205. Springer, 2018, pp. 451–466.

[15] A. Cramer, H. Wu, J. Salamon, and J. P. Bello, “Look, listen, and learn more: Design choices for deep audio embeddings,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3852–3856.

[16] K. Wilkinghoff, “AdaProj: Adaptively scaled angular margin subspace projections for anomaly detection with auxiliary classification tasks,” 2024, arXiv:2403.14179.

[17] X. Zhang, R. Zhao, Y. Qiao, X. Wang, and H. Li, “AdaCos: Adaptively scaling cosine logits for effectively learning deep face representations,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 10 823–10 832.

[18] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “ArcFace: Additive angular margin loss for deep face recognition,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 4690–4699.

[19] K. Wilkinghoff, “Sub-cluster AdaCos: Learning representations for anomalous sound detection,” in *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021.

[20] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[21] H. Lan, Q. Zhu, J. Guan, Y. Wei, and W. Wang, “Hierarchical metadata information constrained self-supervised learning for anomalous sound detection under domain shift,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 7670–7674.

[22] J. Liang, H. Phan, and E. Benetos, “Leveraging label hierachies for few-shot everyday sound recognition,” in *7th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*. Tampere University, 2022.

[23] H. Chen, Y. Song, Z. Zhuo, Y. Zhou, Y.-H. Li, H. Xue, and I. McLoughlin, “An effective anomalous sound detection method based on representation learning with simulated anomalies,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

[24] L. Ruff, N. Görnitz, L. Deecke, S. A. Siddiqui, R. A. Vandermeulen, A. Binder, E. Müller, and M. Kloft, “Deep one-class classification,” in *35th International Conference on Machine Learning (ICML)*, vol. 80. PMLR, 2018, pp. 4390–4399.

[25] K. Wilkinghoff and A. Cornaggia-Urrigshardt, “On choosing decision thresholds for anomalous sound detection in machine condition monitoring,” in *24th International Congress on Acoustics*. The Acoustical Society of Korea, 2022.