MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

The MERL Systems for DCASE 2026 Challenge Task 2

Fujimura, Takuya; Wichern, Gordon; Masuyama, Yoshiki; Boeddeker, Christoph; Saijo, Kohei; Richter, Julius; Edo, Takahiro; Le Roux, Jonathan

TR2026-100 July 01, 2026

Abstract

In this report, we present our anomalous sound detection (ASD) systems for DCASE 2026 Challenge Task 2. Our approach introduces noise-aware audio self-supervised learning (NA-SSL) to leverage two-channel recordings, in which one microphone is used to capture noise. NA-SSL models are trained to extract clean SSL representations from two-channel noisy signals simulated on external datasets. Then, we perform ASD in the extracted denoised feature space. To further improve performance, we perform discrim- inative fine-tuning with attributes and pseudo labels. Furthermore, for anomaly score calculation, we employ several recent techniques: score rescaling, frequency-wise memory bank construction, and deviation-based pooling. Our final ensemble system has achieved 66.20% in the official scores calculated as a harmonic mean of the area under the curve (AUC) and partial AUC ($p = 0.1$) over all machine types and domains in the development set.

IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE Challenge) 2026

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

THE MERL SYSTEMS FOR DCASE 2026 CHALLENGE TASK 2

Technical Report

Takuya Fujimura[1,2*], *Gordon Wichern*[1], *Yoshiki Masuyama*[1], *Christoph Boeddeker*[1], *Kohei Saijo*[3], *Julius Richter*[1], *Takahiro Edo*[1], *Jonathan Le Roux*[1]

[1]Mitsubishi Electric Research Laboratories (MERL), Cambridge, USA,
[2]Nagoya University, Nagoya, Japan,
[3] Information Technology R&D Center, Mitsubishi Electric Corporation, Kanagawa, Japan

ABSTRACT

In this report, we present our anomalous sound detection (ASD) systems for DCASE 2026 Challenge Task 2. Our approach introduces noise-aware audio self-supervised learning (NA-SSL) to leverage two-channel recordings, in which one microphone is used to capture noise. NA-SSL models are trained to extract clean SSL representations from two-channel noisy signals simulated on external datasets. Then, we perform ASD in the extracted denoised feature space. To further improve performance, we perform discriminative fine-tuning with attributes and pseudo labels. Furthermore, for anomaly score calculation, we employ several recent techniques: score rescaling, frequency-wise memory bank construction, and deviation-based pooling. Our final ensemble system has achieved 66.20% in the official scores calculated as a harmonic mean of the area under the curve (AUC) and partial AUC ($p = 0.1$) over all machine types and domains in the development set.

***Index Terms*—** Anomalous sound detection, audio self-supervised learning, noise aware

1. INTRODUCTION

This report describes our systems for DCASE 2026 Challenge Task 2 [1]. This task focuses on anomalous sound detection (ASD), which aims to detect mechanical failures from machine sounds under the following conditions: (i) the training data includes only normal sounds, (ii) there is a data imbalance between the source and target domains, (iii) the machine types differ between the development and evaluation sets, requiring generalization across machine types, and (iv) auxiliary machine attribute information, which is helpful for improving performance, is not always available. These requirements are inherited from recent editions [2, 3]. This year's task additionally introduces two-channel recordings instead of monaural recordings to facilitate system performance improvement [1]. One of the microphones is located close to the target machine, while the other is located farther away to capture predominantly noise.

A recent trend in ASD is the use of self-supervised learning (SSL) models [4–6]. SSL representations effectively capture subtle differences between normal and anomalous sounds, and anomaly scores can be estimated based on the distances between a test sample and normal training samples in the representation space under the condition (i). Although we can further improve the performance by discriminative fine-tuning with attribute labels [5], the original SSL representations have already achieved strong performance [4, 7, 8] without attribute labels. We can also generate pseudo labels from SSL representations to perform discriminative fine-tuning under the unlabeled condition (iv) [9]. Furthermore, SSL-based methods have shown strong performance across various machine types [6], satisfying condition (iii). These properties make SSL-based methods a good fit for this task.

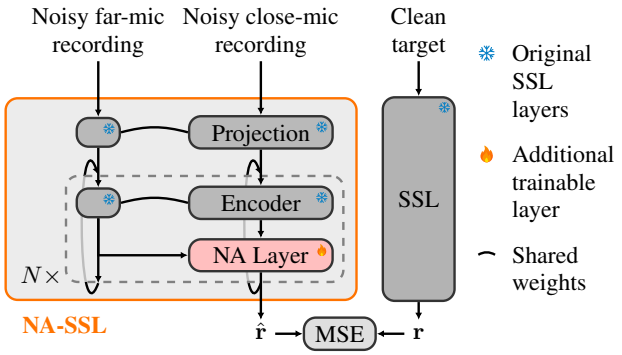

Figure 1: Overview of NA-SSL. The model estimates clean SSL representations $\mathbf{r}$ from two-channel noisy signals, with one microphone placed close to the target sound source and the other placed farther away to capture background noise.

Our solution improves SSL representations using the noise-aware (NA) SSL framework [10]. Specifically, we design NA-BEATs and NA-EAT as the NA extensions of BEATs [11] and EAT [12], respectively. As shown in Fig. 1, these NA-SSL models are trained to extract clean representations from two-channel noisy signals simulated on external datasets. Here, the NA-SSL model learns to extract target sound information dominant in the close microphone by leveraging noise information dominant in the far microphone. To further improve representations for the ASD task, we also perform discriminative fine-tuning using attributes and pseudo labels on the provided machine-sound datasets. For anomaly score calculation, we employ recently proposed frequency-wise memory bank construction [7] and deviation-based pooling [8]. To handle the domain-shift condition (ii), we also employ score rescaling techniques [13–15]. Experimental evaluations on the development dataset show that NA-SSL substantially improves performance, and our final ensemble system achieved an official score of 66.20%, outperforming the official baseline system, which achieved 57.66%.

*This work was done during an internship at MERL.

2. PRIOR WORK: NOISE-AWARE SELF-SUPERVISED LEARNING

NA-SSL aims to extract clean SSL representations from noisy signals by leveraging auxiliary noise information [10]. As a specific realization, NA-BEATs has been proposed based on the widely used BEATs [11], where the auxiliary noise is obtained by cropping a noise-only segment from the noisy input signal [10]. NA-BEATs is trained by distillation with a mean squared error (MSE) loss between the target representation sequence $\mathbf{r} \in \mathbb{R}^{L \times D}$ from the frozen original BEATs and the estimated representation sequence $\hat{\mathbf{r}} \in \mathbb{R}^{L \times D}$, given by

$$\mathbf{r} = \mathrm{BEATs}(\mathbf{s}), \tag{1}$$

$$\hat{\mathbf{r}} = \mathrm{NABEATs}(\mathbf{x}, \mathbf{n}'), \tag{2}$$

where $\mathbf{s}$, $\mathbf{x}$, and $\mathbf{n}'$ denote clean target sound, noisy target sound, and auxiliary noise-only signal, respectively, L is the sequence length, and D is the representation dimension.

The NA-BEATs model is constructed by inserting trainable NA layers into the frozen original BEATs after each Transformer layer. Both the noisy target sound $\mathbf{x}$ and auxiliary noise $\mathbf{n}'$ are passed through the same BEATs layers, and the NA layers refine the representation of the noisy target sound $\mathbf{z}_\mathbf{x} \in \mathbb{R}^{L \times D}$ using the noise representation $\mathbf{z}_{\mathbf{n}'} \in \mathbb{R}^{L \times D}$. Specifically, the cross-attention-based NA layer works as follows:

$$\mathbf{z}_\mathbf{x} \leftarrow \mathbf{z}_\mathbf{x} + \mathrm{MHCA}(\mathrm{Norm}(\mathbf{z}_\mathbf{x}), \mathrm{Norm}(\mathbf{z}_{\mathbf{n}'})), \tag{3}$$

$$\mathbf{z}_\mathbf{x} \leftarrow \mathbf{z}_\mathbf{x} + \mathrm{SwiGLU}(\mathrm{Norm}(\mathbf{z}_\mathbf{x})), \tag{4}$$

where MHCA denotes multi-head cross-attention [16], with the first argument used as the query and the second argument used as the key and value. Norm denotes root mean square normalization [17] and SwiGLU denotes a Swish gated linear unit (SwiGLU)-based feed-forward network (FFN) [18].

NA-BEATs significantly improves performance on various downstream tasks under noisy conditions and generalizes well to unseen noise by using the auxiliary noise information.

3. PROPOSED SYSTEMS

ASD systems consist of a *frontend* and a *backend* [6]. The frontend extracts features from machine sounds, and the backend computes anomaly scores based on distances between the features.

3.1. Frontend

Table 1 summarizes the frontends used in our ASD systems. We use BEATs (*BEATs_iter3.pt*), EAT (*EAT-base_epoch30_pretrain*), and EAT-large (*EAT-large_epoch20_pretrain*) as the base SSL models. All of these models convert mel-spectrogram patches into a representation sequence $\mathbf{r} \in \mathbb{R}^{L \times D}$ through a projection layer and Transformer layers, where L corresponds to the number of mel-spectrogram patches. EAT and EAT-large include an additional CLS token with dimension D to capture global information. These conventional single-channel SSL models take only the close-microphone signal as input.

We design NA-BEATs and NA-EAT as the NA extensions of BEATs and EAT, respectively. We use the same cross-attention-based NA layers and training procedure as described in Section 2, but only modify the auxiliary noise condition to match the challenge setup. In DCASE 2026 Challenge Task 2, the target machine sound is recorded by close and far microphones, while noise is played from the four corners of the room [19]. We carefully simulate this recording setup using Pyroomacoustics [20] and use the far-microphone signal as the auxiliary noise condition. Therefore, the NA-SSL models are trained to extract target sound information that is more dominant in the close microphone than in the far microphone. Also, we obtain the clean target sound at the close microphone with early reflections within 50 ms. Training is performed on large external datasets: FSD50K [21] is used as the target-sound dataset, while WHAM!48kHz [22], DEMAND [23], and QUT-NOISE [24] are used as noise datasets. For NA-EAT, we use MSE losses for the representation sequence and the CLS token with equal weights.

Table 1: SSL models used as frontends in our ASD systems. Dis and NA denote discriminative fine-tuning and NA distillation, respectively. EAT-large was used without discriminative fine-tuning or NA distillation.

Original	NA	Dis	Dis NA
BEATs	NA-BEATs	Dis BEATs	Dis NA-BEATs
EAT	NA-EAT	Dis EAT	Dis NA-EAT
EAT-large	-	-	-

Table 2: Backend techniques used in our ASD systems.

Component	Method
Memory bank	Frequency-wise [7]
Temporal pooling	Average / RDP [8]
Domain shift handling	SMOTE [26] / score rescaling [15]

We also perform discriminative fine-tuning on the provided machine-sound datasets. We aggregate the representation sequence into a single D_{Dis}-dimensional discriminative feature using an attentive statistics pooling layer [25] and a linear layer. The discriminative feature is used only for the classification loss, while the SSL representation sequence before aggregation is used for the backend. For machine types without attribute labels, we generate pseudo labels [9]. The pseudo labels are generated in the source domain, and are then combined with the other available coarse labels of the target machine (i.e., machine-type and domain labels) and attributes of the other machines, to form a multi-class classification task. We first apply temporal pooling [4] to the SSL representations, as follows. First, we reshape the SSL representation sequence $\mathbf{r} \in \mathbb{R}^{L \times D}$ into a time-frequency structure $\mathbf{r}_{\mathrm{TF}} \in \mathbb{R}^{T \times F \times D}$, where T and F are the numbers of mel-spectrogram patches along the time and frequency axes, respectively. Then, we apply average pooling along the time axis and concatenate the frequency axis to obtain a single FD-dimensional feature for each sample. Finally, the pseudo labels are generated by applying k-means clustering to the features within each machine type. When fine-tuning a given SSL frontend, we generate the pseudo labels from that same frontend.

3.2. Backend

Table 2 summarizes the backends used in our ASD systems. First, we employ frequency-wise memory bank construction [7]. Here,

Table 3: Evaluation of frontends using the same backend which consists of frequency-wise memory bank construction, average temporal pooling, and score rescaling. The values are the official scores, and Total is the harmonic mean of the scores over all machine types. * denotes machine types without attribute labels.

Frontend	bearingEmu*	fan	gearboxEmu	sliderEmu*	ToyCar*	ToyCarEmu	valveEmu*	**Total**
BEATs	62.94	52.12	64.43	63.57	**57.40**	57.62	66.59	60.28
EAT	62.74	50.96	61.16	60.71	55.21	**59.24**	67.62	59.24
EAT-large	61.23	51.74	61.09	60.59	55.10	56.73	80.27	59.95
NA-BEATs	62.48	53.15	64.43	65.77	57.26	58.57	93.34	63.18
NA-EAT	62.37	56.58	63.77	66.12	55.73	57.69	90.21	63.13
Dis BEATs	64.07	51.26	66.39	66.52	53.24	57.48	79.28	61.40
Dis EAT	62.84	50.90	63.53	58.63	52.19	59.13	68.67	58.83
Dis NA-BEATs	**65.85**	52.19	**69.44**	69.41	53.45	58.31	**95.21**	63.92
Dis NA-EAT	62.62	**57.69**	68.89	**71.38**	54.25	56.51	92.32	**64.35**

temporal pooling is applied to the time-frequency-structured SSL representation $\mathbf{r}_{\mathrm{TF}}$, and a D-dimensional feature memory bank is constructed for each frequency index. The nearest-neighbor distances between a test sample and normal training samples are calculated separately for each frequency index and then averaged along the frequency axis to obtain the sample-level anomaly score. This technique mitigates undesirably high anomaly scores caused by variations within normal sounds. For example, an unseen combination of machine sound and noise can yield a high anomaly score, even when each frequency component has been separately observed in the training data. By avoiding strict joint comparison across frequencies, this technique improves robustness to normal variations while retaining sensitivity to anomalies in each frequency.

For temporal pooling applied to $\mathbf{r}_{\mathrm{TF}}$ during frequency-wise memory construction, we use either simple average pooling or relative deviation pooling (RDP) [8]. RDP is a weighted average over the representation sequence, where the weights are determined based on the deviation of each representation from the sequence average. This emphasizes representations that deviate from the average, thereby preserving temporally localized anomalies that are smoothed out by typical average pooling. This technique has been observed to be particularly effective with the BEATs frontend [8].

To handle the data imbalance between the source and target domains, we use either SMOTE [26] or score rescaling [14, 15]. SMOTE augments target-domain samples by linear interpolation between nearest neighbors in the feature space. Score rescaling adjusts anomaly scores based on the local density in the feature space and reduces score-scale mismatches between domains caused by data imbalance. Especially, we use the variance-minimization-based score rescaling [15], which further adjusts anomaly scores to minimize their variance in the training data.

4. EXPERIMENTAL EVALUATION

4.1. Setup

We conducted experimental evaluations using the DCASE 2026 Challenge Task 2 development dataset, compiled from ToyADMOS2 [27] and MIMII DG [28], and the additional training dataset. The development dataset includes training and test data for seven machine types: bearingEmu, fan, gearboxEmu, valveEmu, sliderEmu, ToyCarEmu, and ToyCar. The additional training dataset includes training data for five machine types: ToyDrone, ToothBrush, SewingMachine, Sander, and BlowerDustCollector. The training data include 1000 normal samples for each machine type, with 990 samples from the source domain and 10 samples from the target domain. The test data in the development dataset include 50 samples for each machine type, domain, and normal/anomalous class. Attribute labels are available for the following machine types: fan, gearboxEmu, ToyCarEmu, ToyDrone, Sander, and BlowerDustCollector. Each recording was a 6–16-second two-channel signal sampled at 16 kHz.

During training of the NA-SSL models, all signals were resampled to 16 kHz and randomly cropped or zero-padded to 10 s. For impulse response simulation with Pyroomacoustics, the room length and width were randomly sampled from [3.0, 8.0] m, and the height was sampled from [1.5, 3.0] m. The reverberation time, RT60, was randomly sampled from [0.1, 0.4] s. The target sound source was randomly placed with a 0.5 m margin from the walls. The close microphone was fixed 0.05 m from the target source, while the far microphone was randomly placed [0.1, 1.0] m away from the close microphone. Four different noise sources were randomly selected and placed 0.2 m away from the room corners. All sound sources and microphones were placed at a height of 1.0 m. The SNR at the close microphone was randomly selected from [-10, 10] dB.

For the NA layers, we set the number of heads in MHCA to 8 and the hidden size of the FFN to 2304. We trained both NA-BEATs and NA-EAT for 200 epochs using the AdamW optimizer, a fixed learning rate of 0.0001, and a batch size of 160. We applied exponential moving average (EMA) with a decay rate of 0.999 after 20k training steps.

For discriminative fine-tuning, we applied low-rank adaptation (LoRA) with a rank of 64 to the query, key, and value projection layers in the attention modules. For NA-SSL, we also applied LoRA to MHCA in NA layers. We fine-tuned the models for 25 epochs using sub-cluster AdaCos [29] with 16 sub-clusters, $D_{\mathrm{Dis}} = 256$, mixup with a probability of 0.5, AdamW optimizer, and a batch size of 8. The learning rate was linearly increased from 0 to 0.0001 over the first 5,000 steps.

For RDP in the backend, we set the deviation power γ to 4. For score rescaling, we set the number of nearest neighbors to 4. For SMOTE, we augmented the target-domain samples to 20% of the source-domain samples using two nearest neighbors.

As the evaluation metric, we used the official score [1] calculated as the harmonic mean of three types of area under the

Table 4: Configuration of the final systems. The backend always employs frequency-wise memory bank construction but employs different techniques for temporal pooling and domain-shift handling. *Only Fujimura_MERL_task2_2 used weighted averaging, where the weights for BEATs, EAT, EAT-large, NA-EAT, and NA-BEATs are 1, 0.5, 0.5, 1, and 1, respectively. †Fujimura_MERL_task2_4 selected Dis NA-EAT when attribute labels are available, and selected NA-EAT otherwise.

System name	Frontend set	Temporal pooling	Domain shift handling	Ensemble method
Fujimura_MERL_task2_1	Dis NA-EAT, Dis NA-BEATs	Average	SMOTE	Averaging anomaly scores
Fujimura_MERL_task2_2	BEATs, EAT, EAT-large, NA-EAT, NA-BEATs	Average	Score rescaling	Averaging anomaly scores*
Fujimura_MERL_task2_3	Dis NA-BEATs	RDP	Score rescaling	N/A (single frontend)
Fujimura_MERL_task2_4	NA-EAT, Dis NA-EAT	Average	Score rescaling	Selection based on label availability†

Table 5: Evaluation results of the baseline systems and the final systems. The values are the official scores, and Total is the harmonic mean of the scores over all machine types. * denotes machine types without attribute labels.

System name	bearingEmu*	fan	gearboxEmu	sliderEmu*	ToyCar*	ToyCarEmu	valveEmu*	**Total**
Baseline (MSE)	60.56	53.26	55.93	52.71	51.60	61.73	63.22	56.66
Baseline (MAHALA)	62.79	51.75	58.92	54.29	61.33	62.37	54.26	57.66
Fujimura_MERL_task2_1	59.15	55.86	65.61	68.69	**70.71**	**63.11**	89.70	**66.20**
Fujimura_MERL_task2_2	62.76	55.36	64.19	66.38	57.11	58.19	90.73	63.43
Fujimura_MERL_task2_3	**65.28**	52.95	**69.66**	**73.19**	54.14	57.80	**95.99**	64.57
Fujimura_MERL_task2_4	62.37	**57.69**	68.89	66.12	55.73	56.51	90.21	63.79

receiver operating characteristic (ROC) curve (AUC): (a) partial AUC (pAUC) with $p = 0.1$, calculated using samples from both the source and target domains; and (b) source-domain and (c) target-domain AUCs, each calculated using normal samples from the corresponding domain and anomalous samples from both domains.

We performed NA-SSL training and discriminative fine-tuning for three trials with different random seeds and averaged the anomaly scores across the trials.

4.2. Results

Table 3 compares frontends using the same backend consisting of frequency-wise memory bank construction, average temporal pooling, and score rescaling. First, we can see that NA-SSL is effective regardless of the base model (i.e., BEATs or EAT), and whether discriminative fine-tuning is applied or not. Both NA-BEATs and NA-EAT show substantial improvements of over 20 % for valveEmu, and each also improves performance for other machine types. Discriminative fine-tuning further improves the performance, and the top two total scores are obtained by Dis NA-EAT and Dis NA-BEATs. Also, by using pseudo labels, the benefits of discriminative fine-tuning can be observed even for machine types without attribute labels (e.g., bearingEmu, sliderEmu, and valveEmu).

Table 4 shows the configurations of our final systems. The final systems ensemble multiple frontends with a single backend, where the backend differs across systems. The ensemble of frontends is performed by averaging their anomaly scores or by selecting models with or without discriminative fine-tuning based on the availability of attribute labels. Although discriminative fine-tuning is generally effective, it can sometimes degrade performance [9]. Therefore, we also submit Fujimura_MERL_task2_2 as a system without fine-tuning.

Table 5 shows the evaluation results of our final systems and the official baseline systems [30]. The baseline systems are based on autoencoders, and two variants, MSE and MAHALA, are provided. All of our systems substantially outperform these official baseline systems. The best official score of the baseline systems is 57.66%, while our best system, Fujimura_MERL_task2_1, achieves 66.20%. In addition, the effectiveness of RDP can be observed by comparing Dis NA-BEATs in Table 3 with Fujimura_MERL_task2_3 in Table 5. The former uses average pooling and achieves a total score of 63.92%, whereas the latter uses RDP and improves the score to 64.57%.

5. CONCLUSION

In this report, we presented our systems for DCASE 2026 Challenge Task 2. We introduced NA-SSL to leverage two-channel recordings, where one microphone is located far from the target machine to capture noise. Specifically, we designed NA-BEATs and NA-EAT as NA extensions of BEATs and EAT, respectively. We also employed several techniques to further improve performance, namely discriminative fine-tuning, pseudo labeling, frequency-wise memory bank construction, RDP, and score rescaling. Experimental results on the development set demonstrated the consistent effectiveness of NA-SSL, regardless of the base SSL model and whether discriminative fine-tuning was applied. One of our final systems achieved an official score of 66.20%, substantially outperforming the official baseline systems, which achieved 57.66%.

6. REFERENCES

[1] T. Nishida, N. Harada, D. Takeuchi, D. Niizumi, K. Imoto, K. Dohi, H. Purohit, T. Endo, and Y. Kawaguchi, "Description and discussion on DCASE 2026 challenge task 2: Noise-aware unsupervised anomalous sound detection for machine condition monitoring," *In arXiv e-prints: 2606.01578*, 2026.

[2] T. Nishida et al., "Description and discussion on DCASE 2024 challenge task 2: First-shot unsupervised anomalous sound detection for machine condition monitoring," in *Proc. DCASE*, 2024.

[3] T. Nishida et al., "Description and discussion on DCASE 2025 challenge task 2: First-shot unsupervised anomalous sound detection for machine condition monitoring," in *Proc. DCASE*, 2025.

[4] P. Saengthong and T. Shinozaki, "Deep generic representations for domain-generalized anomalous sound detection," in *Proc. ICASSP*, 2025.

[5] A. Jiang, B. Han, Z. Lv, Y. Deng, W.-Q. Zhang, X. Chen, Y. Qian, J. Liu, and P. Fan, "Anopatch: Towards better consistency in machine anomalous sound detection," in *Proc. Interspeech*, 2024.

[6] T. Fujimura, K. Wilkinghoff, K. Imoto, and T. Toda, "ASDKit: A toolkit for comprehensive evaluation of anomalous sound detection methods," in *Proc. DCASE*, 2025.

[7] P. Saengthong and T. Shinozaki, "Sub-band spectral matching with localized score aggregation for robust anomalous sound detection," *arXiv preprint arXiv:2603.13749*, 2026.

[8] K. Wilkinghoff, S. Yadav, and Z.-H. Tan, "Temporal pooling strategies for training-free anomalous sound detection with self-supervised audio embeddings," *arXiv preprint arXiv:2603.04605*, 2026.

[9] T. Fujimura, I. Kuroyanagi, and T. Toda, "Improvements of discriminative feature space training for anomalous sound detection in unlabeled conditions," in *Proc. ICASSP*, 2025.

[10] T. Fujimura, Y. Masuyama, G. Wichern, C. Boeddeker, J. Richter, and J. Le Roux, "NABEATs: Noise-aware audio representation learning," in *Proc. IWAENC*, (submitted), 2026.

[11] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, W. Che, X. Yu, and F. Wei, "BEATs: Audio pre-training with acoustic tokenizers," in *Proc. ICML*, 2023.

[12] W. Chen, Y. Liang, Z. Ma, Z. Zheng, and X. Chen, "EAT: Self-supervised pre-training with efficient audio transformer," in *Proc. IJCAI*, Main Track, 2024.

[13] K. Wilkinghoff, T. Fujimura, K. Imoto, J. Le Roux, Z.-H. Tan, and T. Toda, "Handling domain shifts for anomalous sound detection: A review of DCASE-related work," in *Proc. DCASE*, 2025.

[14] K. Wilkinghoff, H. Yang, J. Ebbers, F. G. Germain, G. Wichern, and J. Le Roux, "Local density-based anomaly score normalization for domain generalization," *IEEE/ACM TASLP*, vol. 33, pp. 4642–4652, 2025.

[15] M. Matsumoto, T. Fujimura, W. Huang, and T. Toda, "Adjusting bias in anomaly scores via variance minimization for domain-generalized discriminative anomalous sound detection," in *Proc. DCASE*, 2025.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NeurIPS*, vol. 30, 2017.

[17] B. Zhang and R. Sennrich, "Root mean square layer normalization," in *Proc. NeurIPS*, vol. 32, 2019.

[18] N. Shazeer, "GLU variants improve transformer," *arXiv preprint arXiv:2002.05202*, 2020.

[19] DCASE Community, *DCASE 2026 challenge task 2: Noise-aware unsupervised anomalous sound detection for machine condition monitoring*, https://dcase.community/challenge2026/task-first-shot-unsupervised-anomalous-sound-detection-for-machine-condition-monitoring, Accessed: 2026-06-17, 2026.

[20] R. Scheibler, E. Bezzam, and I. Dokmanić, "Pyroomacoustics: A python package for audio room simulation and array processing algorithms," in *Proc. ICASSP*, 2018.

[21] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: An open dataset of human-labeled sound events," *IEEE/ACM TASLP*, vol. 30, pp. 829–852, 2022.

[22] G. Wichern, J. Antognini, M. Flynn, L. R. Zhu, E. McQuinn, D. Crow, E. Manilow, and J. Le Roux, "WHAM!: Extending speech separation to noisy environments," in *Proc. Interspeech*, 2019.

[23] J. Thiemann, N. Ito, and E. Vincent, "The Diverse Environments Multi-channel Acoustic Noise Database (DEMAND): A database of multichannel environmental noise recordings," in *POMA*, vol. 19, 2013.

[24] D. Dean, S. Sridharan, R. Vogt, and M. Mason, "The QUT-NOISE-TIMIT corpus for the evaluation of voice activity detection algorithms," in *Proc. Interspeech*, 2010.

[25] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," in *Proc. Interspeech*, 2018.

[26] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *JAIR*, vol. 16, pp. 321–357, 2002.

[27] 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 *Proc. DCASE*, 2021.

[28] 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 *Proc. DCASE*, 2022.

[29] K. Wilkinghoff, "Sub-cluster AdaCos: Learning representations for anomalous sound detection," in *Proc. IJCNN*, 2021.

[30] N. Harada, D. Niizumi, Y. Ohishi, D. Takeuchi, and M. Yasuda, "First-shot anomaly sound detection for machine condition monitoring: A domain generalization baseline," in *Proc. EUSIPCO*, 2023.