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

NABEATs: Noise-Aware Audio Representation Learning

Fujimura, Takuya; Masuyama, Yoshiki; Wichern, Gordon; Boeddeker, Christoph; Richter, Julius; Le Roux, Jonathan

TR2026-124 September 09, 2026

Abstract

We propose the concept of noise-aware audio self-supervised learning (SSL), whose goal is to encode audio mixtures while suppressing undesired noise, and present Noise-Aware BEATs (NABEATs) as a BEATs-based realization of this framework. Audio SSL models are designed to handle a wide range of audio signals. Consequently, under noisy conditions, they cannot effectively focus on the target sounds relevant to a downstream task, resulting in degraded performance. To address this issue, NABEATs is trained to estimate clean BEATs representations from a noisy audio signal with an auxiliary reference noise input. This reference noise enables the model to account for specific noise characteristics at inference time, thereby achieving better generalization across operating environments. Our experimental evaluations demonstrate that NABEATs significantly improves performance of various downstream tasks under noisy conditions and also generalizes well to unseen noise types.

International Workshop on Acoustic Signal Enhancement (IWAENC) 2026

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

NABEATS: NOISE-AWARE AUDIO REPRESENTATION LEARNING

Takuya Fujimura[1,2*], *Yoshiki Masuyama*[1], *Gordon Wichern*[1],
Christoph Boeddeker[1], *Julius Richter*[1], *Jonathan Le Roux*[1]

[1]Mitsubishi Electric Research Laboratories (MERL), Cambridge, USA,
[2]Nagoya University, Nagoya, Japan

ABSTRACT

We propose the concept of noise-aware audio self-supervised learning (SSL), whose goal is to encode audio mixtures while suppressing undesired noise, and present Noise-Aware BEATs (NABEATs) as a BEATs-based realization of this framework. Audio SSL models are designed to handle a wide range of audio signals. Consequently, under noisy conditions, they cannot effectively focus on the target sounds relevant to a downstream task, resulting in degraded performance. To address this issue, NABEATs is trained to estimate clean BEATs representations from a noisy audio signal with an auxiliary reference noise input. This reference noise enables the model to account for specific noise characteristics at inference time, thereby achieving better generalization across operating environments. Our experimental evaluations demonstrate that NABEATs significantly improves performance of various downstream tasks under noisy conditions and also generalizes well to unseen noise types.

Index Terms— self-supervised learning, general-purpose audio representation, noise robustness

1. INTRODUCTION

Self-supervised learning (SSL) models have achieved remarkable success in speech and audio processing. These models learn general-purpose representations from large amounts of unlabeled audio data, enabling transfer to various downstream tasks. For example, speech SSL models [1, 2] have demonstrated impressive performance in downstream tasks such as automatic speech recognition, even with a small amount of labeled data or a simple model [3]. Beyond speech, audio SSL models [4–7] have also shown their effectiveness across various downstream tasks including the environmental sound and music domains.

Inspired by the success of target sound extraction (TSE) [8–10], SSL models that extract target representations from noisy signals have also been explored, especially in the speech domain [11–13]. A representative example is WavLM [11], which is trained to predict pseudo labels of the primary speaker's utterance and ignore both noise and non-primary speech. In addition, speaker-aware variants [14–16] have also been proposed to extract the representation of a target speaker specified by a reference enrollment utterance. These methods have made SSL models more effective under real-world noisy conditions and have broadened their applicability.

Despite the progress in noise-robust SSL models for speech, this direction has not yet been explored for general audio SSL models. In addition, audio SSL models are expected to be applicable to a wider variety of downstream tasks than speech SSL models. A method in the spirit of WavLM would thus not be practical, as some types of sounds may need to be preserved for some downstream tasks but suppressed in others. Furthermore, unlike speaker-aware SSL, relying on target sound information as reference is unrealistic for some downstream tasks: for example, in audio tagging, which aims to identify the sound class, providing the target sound is essentially equivalent to solving the task. In contrast, even when the target sound is unknown, noise information can often be available, and using noise as a reference is more practical in real-world scenarios.

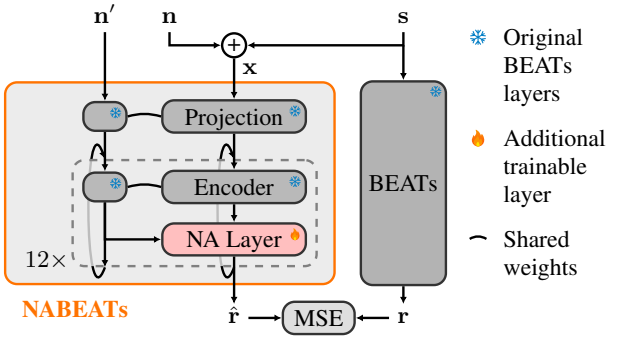

Fig. 1: Overview of NABEATs. The model estimates clean BEATs representations $\mathbf{r}$ from noisy inputs $\mathbf{x}$ by leveraging an additional reference noise signal $\mathbf{n}'$ for conditional denoising.

In this paper, we propose the concept of noise-aware audio SSL, whose goal is to encode audio mixtures while suppressing undesired noise similar to a given reference. As a specific realization, we introduce noise-aware BEATs (NABEATs), built upon the widely used and high-performing BEATs [4] model. NABEATs is trained to estimate clean BEATs representations from noisy audio signals. To achieve this, the model takes as additional input a reference noise signal $\mathbf{n}'$ sharing characteristics with the noise $\mathbf{n}$ in the noisy input (see Fig. 1) and learns to perform conditional denoising. We also introduce denoising BEATs (DBEATs) as a standard denoising baseline, which does not use the reference and is trained similarly to WavLM [11]. In the experimental evaluation, we demonstrate that both NABEATs and DBEATs significantly improve performance under noisy conditions across various downstream tasks, while NABEATs shows superior generalization to unseen noise.

2. PROPOSED METHODS

The intended usage scenario of NABEATs is similar to that of traditional spectral subtraction [17], where a reference noise signal is obtained from noise-only segments, pre-recorded noise at the target location, or a separately placed microphone. In this paper, we focus

*This work was done during an internship at MERL.

on the case where the reference noise is obtained from a different time segment from the same recording. We consider this setting to be a practical and reasonable choice, and investigating other conditions is left to future work. The proposed method is expected to handle non-stationary noise better than spectral subtraction, and avoids cascaded systems by operating directly in the representation space.

Our base model, BEATs, is an audio SSL model that performs masked prediction in the discrete token space, where the tokens are obtained from mel-spectrogram patches using an acoustic tokenizer. BEATs extracts a representation sequence $\mathbf{r} \in \mathbb{R}^{L\times 768}$ from mel-spectrogram patches through a projection layer and 12 Transformer encoder layers, where L is the sequence length equal to the number of mel-spectrogram patches. BEATs is trained to predict the correct tokens in the masked positions from $\mathbf{r}$, and the tokenizer is also iteratively updated through distillation from the BEATs model, leading to strong downstream performance.

Figure 1 shows the training framework of NABEATs. It is trained by distillation using the following mean squared error (MSE) loss $\mathcal{L}_{\mathrm{MSE}}$ between the target representation $\mathbf{r}$ from the frozen original BEATs and its estimate $\hat{\mathbf{r}}$ from NABEATs, given by

$$\mathcal{L}_{\mathrm{MSE}} = \frac{1}{L\cdot 768}\left\|\mathbf{r}-\hat{\mathbf{r}}\right\|_F^2, \tag{1}$$

$$\mathbf{r} = \mathrm{BEATs}(\mathbf{s}), \tag{2}$$

$$\hat{\mathbf{r}} = \mathrm{NABEATs}(\mathbf{x}, \mathbf{n}'), \tag{3}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. Here, $\mathbf{s}$ and $\mathbf{x} = \mathbf{s} + \mathbf{n}$ denote target sound and noisy input sound, respectively, and $\mathbf{n}'$ is a reference noise as introduced in Section 1. We also introduce DBEATs as a standard denoising baseline, which does not use the reference noise:

$$\hat{\mathbf{r}} = \mathrm{DBEATs}(\mathbf{x}). \tag{4}$$

The training procedure of DBEATs is identical to that of NABEATs.

Following [18], we construct DBEATs and NABEATs by inserting additional layers after every encoder layer of the original BEATs, and train only these layers while keeping the original BEATs components frozen. For DBEATs, we employ the following standard Transformer layer as the additional denoising layer:

$$\mathbf{z}_{\mathbf{x}} \leftarrow \mathbf{z}_{\mathbf{x}} + \mathrm{MHSA}(\mathrm{RMSNorm}(\mathbf{z}_{\mathbf{x}})), \tag{5}$$

$$\mathbf{z}_{\mathbf{x}} \leftarrow \mathbf{z}_{\mathbf{x}} + \mathrm{FFN}_{\mathrm{SwiGLU}}(\mathrm{RMSNorm}(\mathbf{z}_{\mathbf{x}})), \tag{6}$$

where $\mathbf{z}_{\mathbf{x}} \in \mathbb{R}^{L\times 768}$ is the representation of the noisy input $\mathbf{x}$ from the previous layer, MHSA denotes multi-head self-attention [19], RMSNorm denotes root mean square normalization [20], and $\mathrm{FFN}_{\mathrm{SwiGLU}}$ denotes a Swish gated linear unit (SwiGLU)-based feed-forward network (FFN) [21]. For NABEATs, both $\mathbf{x}$ and $\mathbf{n}'$ are passed through the same frozen BEATs components, and the additional noise-aware (NA) layers refine the representation $\mathbf{z}_{\mathbf{x}}$ by leveraging the reference noise representation. We consider two types of NA layers, based on cross-attention (CA) and FiLM [22], and refer to the resulting models as NABEATs-CA and NABEATs-FiLM, respectively. Both models replace only (5) with CA or FiLM while keeping the same FFN as in (6). Specifically, NABEATs-CA employs the following CA instead of (5):

$$\mathbf{z}_{\mathbf{x}} \leftarrow \mathbf{z}_{\mathbf{x}} + \mathrm{MHCA}(\mathrm{RMSNorm}\left(\mathbf{z}_{\mathbf{x}}\right), \mathrm{RMSNorm}\left(\mathbf{z}_{\mathbf{n}'}\right)), \tag{7}$$

where MHCA denotes multi-head CA, with the first argument used as the query and the second argument used as the key and value, and $\mathbf{z}_{\mathbf{n}'}$ is the representation of the reference noise $\mathbf{n}'$. NABEATs-FiLM applies the same feature-wise linear modulation to each element of the representation sequence, modifying the l-th element $\mathbf{z}_{\mathbf{x}}^l$ using the averaged reference noise representation $\bar{\mathbf{z}}_{\mathbf{n}'}$:

$$\bar{\mathbf{z}}_{\mathbf{n}'} = \frac{1}{L}\sum_{l=1}^{L}\mathrm{RMSNorm}(\mathbf{z}_{\mathbf{n}'}^l), \tag{8}$$

$$\boldsymbol{\gamma} = \mathrm{Affine}(\bar{\mathbf{z}}_{\mathbf{n}'}) \in \mathbb{R}^{768}, \tag{9}$$

$$\boldsymbol{\beta} = \mathrm{Affine}(\bar{\mathbf{z}}_{\mathbf{n}'}) \in \mathbb{R}^{768}, \tag{10}$$

$$\mathbf{z}_{\mathbf{x}}^l \leftarrow \mathbf{z}_{\mathbf{x}}^l + \boldsymbol{\gamma}\odot\mathrm{RMSNorm}(\mathbf{z}_{\mathbf{x}}^l) + \boldsymbol{\beta}, \tag{11}$$

where Affine denotes an affine layer, and $\odot$ denotes element-wise multiplication. NABEATs-FiLM uses only global reference-noise information, whereas NABEATs-CA adaptively aggregates reference information across the sequence.

Another way to implement noise-aware SSL is to use a cascaded architecture that first performs TSE and then feeds its output into the SSL model. However, such a cascaded approach tends to be inefficient and incur higher computational cost [23]; therefore, in this paper, we focus on an integrated SSL model.

Table 1: Datasets used for the training of DBEATs and NABEATs. Text in parentheses indicates the official split of each dataset used at each stage.

Stage	Target	Noise
Train	FSD50K (Train)	WHAM! (Train), DEMAND, QUT-NOISE
Valid	FSD50K (Valid)	WHAM! (Valid)

Table 2: Noise datasets used for downstream evaluation. For WHAM! and MUSDB, we followed the official splits. The CHiME-3 row shows the recording IDs used for each split.

Noise dataset	Train	Valid	Test
WHAM! (seen)	Train	Valid	Test
CHiME-3 (unseen)	040 & 050	030	010 & 020
MUSDB18 (unseen)	Train	Valid	Test

3. EXPERIMENTAL SETUP

We conduct two types of evaluations. First, in Section 4, we evaluate performance on various downstream tasks under simulated noisy conditions using a custom-designed setup. For the reference noise $\mathbf{n}'$ in NABEATs, we use a noise-only segment immediately preceding $\mathbf{n}$ from the same recording, with the same duration, for both NABEATs training and the downstream tasks. Second, in Section 5, we evaluate on the DCASE 2025 Task 2 [24], which addresses anomalous sound detection (ASD) for machine condition monitoring. In this task, the dataset contains noisy machine sounds and separately recorded noise-only signals, aligning well with one of the intended applications of our NABEATs framework.

Table 1 summarizes the datasets for the training of NABEATs and DBEATs. We used FSD50K [25] to sample the target sound. Although BEATs itself is trained on the larger Audioset [26], we leave such large-scale training to future work. As the noise dataset, we used WHAM!48kHz [27], following the official splits. In addition, we used DEMAND [28] and QUT-NOISE [29] as training noise datasets. All of these noise datasets contain environmental sounds

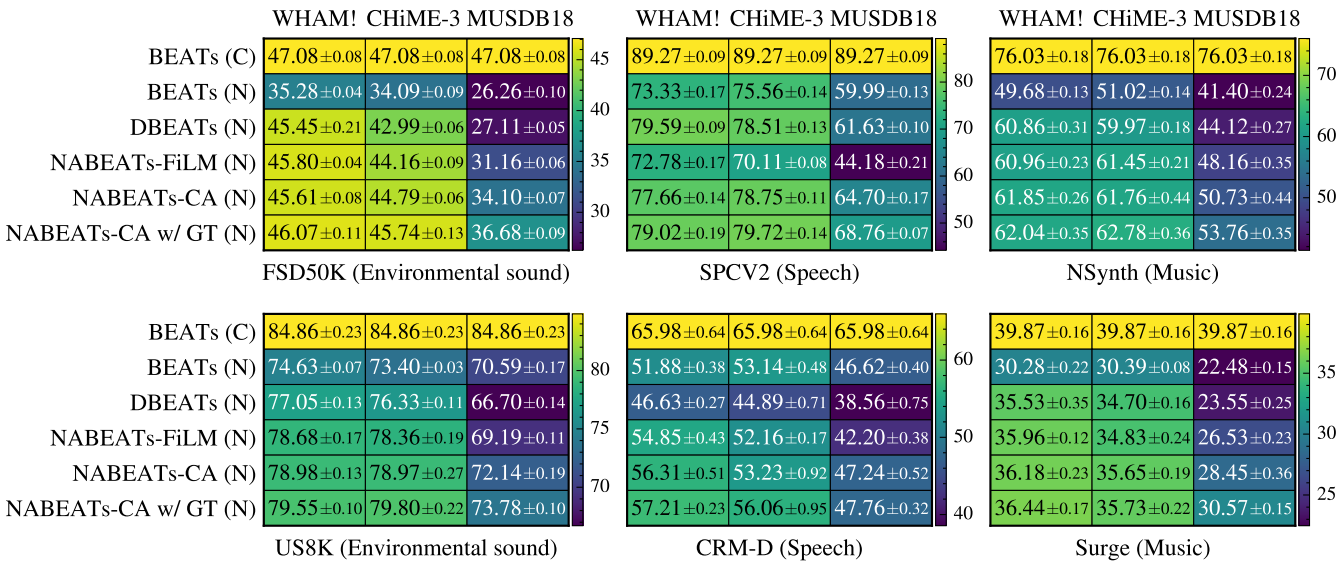

Fig. 2: Downstream classification performance under noisy conditions, where mAP is used for FSD50K and accuracy is used for the other tasks. Each panel represents one task, with the columns corresponding to different noise conditions and the rows to different models. (C) and (N) denote clean and noisy input, respectively. Average scores across six trials with different random seeds are shown, together with 95% confidence intervals. "w/ GT" indicates the performance when the reference noise for NABEATs is the ground-truth noise $\mathbf{n}$.

such as traffic, café, or park noise. In our experiments, all audio signals were resampled to 16 kHz. For both training and validation, we mixed a target sound with a noise signal at random SNRs between -5 and 10 dB. During training, we randomly cropped or zero-padded the target sound to 10 sec.

For BEATs, we used the *BEATs_iter3.pt* checkpoint.[1] In the additional layers in DBEATs and NABEATs, we used MHSA and MHCA with 8 heads, respectively. For FFN, we set the hidden size to 2304. We trained both DBEATs and NABEATs for 250 epochs using the AdamW optimizer, a fixed learning rate of 0.0001, and a batch size of 80. We applied an exponential moving average with a decay rate of 0.999 after 20k training steps, and evaluated the model with the best validation loss.

4. EVALUATION ON VARIOUS DOWNSTREAM TASKS

4.1. Setups

We conducted evaluation on various downstream tasks according to [5, 6]. We trained only a single linear layer on top of the frozen SSL model for downstream classification tasks. We adopted six downstream tasks covering environmental sound, speech, and music domains. For environmental sound classification, we used FSD50K and UrbanSound8K (US8K) [30]. For speech-related tasks, we used Speech Commands V2 (SPCV2) [31] for word classification and CREMA-D (CRM-D) [32] for emotion recognition. For music tasks, we used NSynth [33] for instrument family classification and the Surge Pitch Dataset (Surge) [34] for pitch classification.

For US8K, we conducted 10-fold cross-validation following the official splits. During cross-validation, we randomly divided the non-test folds into training and validation sets with a ratio of 10:1. For the other datasets, we followed the official training, validation, and test splits. Following [5, 6], for each dataset, audio signals were cropped or zero-padded to the average duration of the dataset. For metrics, we used mean average precision (mAP) for FSD50K, a multi-label task, and accuracy for all other single-label tasks [5, 6].

Table 2 summarizes the noise datasets used to simulate noisy conditions for the above downstream tasks. In addition to WHAM!, which was used to train NABEATs and DBEATs, we used CHiME-3 [35] and MUSDB18 [36] to evaluate generalization to unseen noise types. CHiME-3 contains environmental noise similar to the noise datasets used to train NABEATs and DBEATs, whereas MUSDB18 contains music signals that are substantially different from them. We split CHiME-3 based on the recording IDs. For MUSDB18, we used the official dataset parser[2] to obtain the validation set. For WHAM! and MUSDB18, we excluded signals shorter than 60 sec to ensure that reference noise can be obtained from the same recording across all downstream tasks, while CHiME-3 contains only long recordings. For each split of the downstream datasets, we mixed each sample with a noise signal from the corresponding split of the noise dataset at random SNRs between -5 and 10 dB.

In the downstream tasks, we fed each representation averaged across the sequence into a linear layer [4], which we trained for up to 200 epochs using the Adam optimizer, a learning rate of 0.0003, and a batch size of 200. We used the validation set for early stopping with a patience of 20 epochs. We conducted each evaluation six times with different random seeds, and report the average results with 95% confidence intervals.

4.2. Results

Figure 2 shows the results across various downstream tasks under noisy conditions, where we used noisy signals as input during both training and inference of the linear head. First, we can see that performance of the original BEATs degrades significantly under noisy conditions compared with clean conditions in all tasks. In contrast, NABEATs-CA and DBEATs improve the performance in the WHAM! and CHiME-3 noise conditions, except for DBEATs on

[1] https://github.com/microsoft/unilm/tree/master/beats

[2] https://github.com/sigsep/sigsep-mus-db

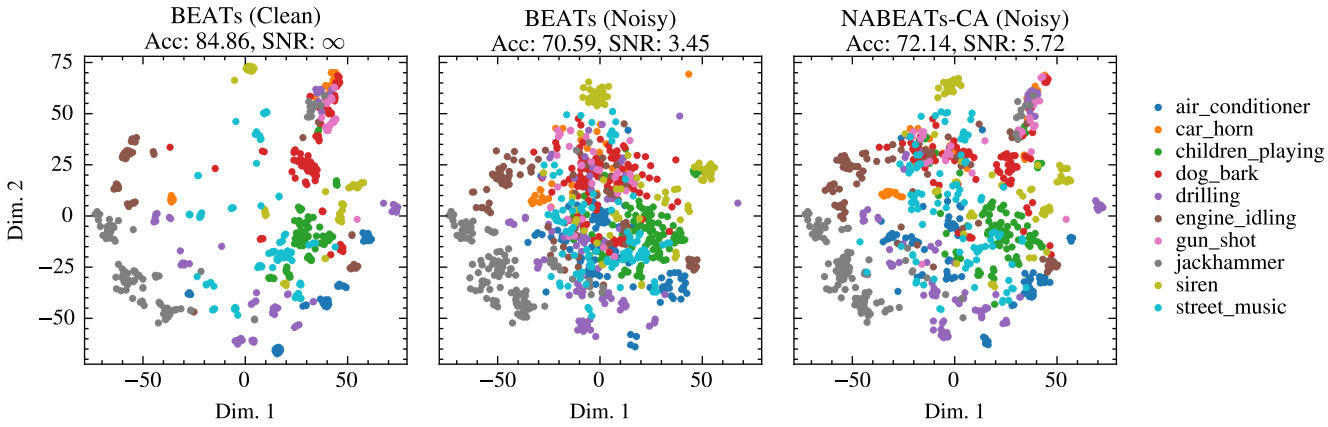

Fig. 3: t-SNE visualization of the averaged representation of the test set for US8K under the clean condition for BEATs and under the MUSDB18 noise condition for BEATs and NABEATs-CA. All figures have the same axes.

CRM-D. In the MUSDB18 noise condition, while DBEATs suffers from a large mismatch with the training noise data, NABEATs-CA still improves performance, demonstrating its better generalization to unseen noise conditions. Furthermore, by leveraging reference noise, NABEATs-CA achieves better performance than DBEATs even in seen noise conditions in all tasks except SPCV2. From the comparison between NABEATs-CA and NABEATs-FiLM, we can see that NABEATs-CA performs particularly well on the speech and music tasks and under the MUSDB18 noise condition, indicating that these tasks require more effective handling of time-varying information than environmental sound tasks. Lastly, to investigate the behavior of NABEATs, we evaluated NABEATs-CA w/ GT, which uses the ground-truth noise $\mathbf{n}$ as the reference noise. It yields a slight but consistent performance improvement over NABEATs-CA, which can be regarded as a reasonable upper-bound performance. Since NABEATs is not trained using the ground-truth noise $\mathbf{n}$ as the reference noise, this result also shows the robustness of NABEATs to a different reference noise segment.

Table 3 shows the signal-to-noise ratio (SNR) between the clean representation sequences $\mathbf{r}$ and their estimates $\hat{\mathbf{r}}$, highlighting the substantial improvement by the proposed methods. We note that DBEATs improves SNR even on CRM-D, despite degrading downstream performance, which indicates a potential mismatch between downstream performance and the MSE objective. We leave investigation of better training objectives to future work.

Figure 3 shows t-SNE [37] visualizations of the averaged representation of the test set for US8K under the clean condition for BEATs and under the MUSDB18 noise condition for BEATs and NABEATs-CA. The original BEATs representations are well separated across classes in the clean condition, but become mixed under the noisy condition. In contrast, NABEATs-CA maintains better class separability under noisy conditions, leading to better downstream performance and SNR.

5. EVALUATION ON DCASE 2025 TASK 2

We also conducted evaluation using the DCASE 2025 Task 2 dataset [24], compiled from [38–40]. This dataset contains 15 machine types, each with 1000 normal training samples and 200 test samples including normal and anomalous samples. Nine of the 15 machine types include 100 noise-only signals as supplementary data, and we restrict our experiments to these machine types.

Following [41], without using any additional neural network, we simply calculated the anomaly score based on the distance between the representation of the test sample and those of the normal training samples, where we averaged the representation across the sequence. We used the standard anomaly score calculation method which uses SMOTE oversampling [42] to mitigate the class imbalance in the training set and then calculates the cosine distance to the nearest training sample [43, 44]. For NABEATs, we selected a reference noise sample from the supplementary data with minimum distance to the input noisy machine sound in the original BEATs representation.

As the evaluation metric, we used the official task score based on the area under the receiver operating characteristic curve [24] and report its harmonic mean across the nine machine types.

The evaluation scores on DCASE 2025 Task 2 are 54.57 for BEATs, 55.76 for DBEATs, and **56.15** for NABEATs-CA. This further verifies the effectiveness of the proposed methods even when the reference noise is not the segment immediately preceding $\mathbf{n}$, and also demonstrates their effectiveness on a public benchmark dataset.

Table 3: SNR [dB] results between the clean and estimated representation sequences. W and M indicate WHAM! and MUSDB18 noise conditions, respectively.

	FSD50K		CRM-D		NSynth	
	W	M	W	M	W	M
BEATs	2.0	1.3	3.2	2.4	0.2	0.0
DBEATs	7.5	3.3	3.8	2.2	4.4	1.4
NABEATs-CA	**7.6**	**5.4**	**5.5**	**4.1**	**4.5**	**3.2**

6. CONCLUSION

In this paper, we explored the concept of noise-aware audio SSL and proposed NABEATs as a realization of this framework based on BEATs. While the standard denoising baseline DBEATs is simply trained to estimate clean representations from noisy signals, NABEATs additionally takes a reference noise as input and is trained to suppress the corresponding noise information. Our experiments showed that, while both methods significantly improve performance under noisy conditions, NABEATs outperforms DBEATs in both seen and unseen noise conditions, highlighting the benefit of using a reference noise to make the model noise-aware. Future work includes investigating other reference noise settings, better training objectives for downstream tasks, and larger-scale training.

7. REFERENCES

[1] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM TASLP*, vol. 29, pp. 3451–3460, 2021.

[2] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. NeurIPS*, vol. 33, 2020.

[3] S.-w. Yang et al., "SUPERB: Speech Processing Universal PERformance Benchmark," in *Proc. Interspeech*, 2021.

[4] 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.

[5] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "BYOL for Audio: Exploring pre-trained general-purpose audio representations," *IEEE/ACM TASLP*, vol. 31, pp. 137–151, 2022.

[6] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "Masked Modeling Duo: Towards a universal audio pre-training framework," *IEEE/ACM TASLP*, vol. 32, pp. 2391–2406, 2024.

[7] Y. Gong, C.-I. Lai, Y.-A. Chung, and J. Glass, "SSAST: Self-supervised audio spectrogram transformer," in *Proc. AAAI*, vol. 36, 2022.

[8] M. Delcroix, K. Zmolikova, K. Kinoshita, A. Ogawa, and T. Nakatani, "Single channel target speaker extraction and recognition with speaker beam," in *Proc. ICASSP*, 2018.

[9] X. Liu, Q. Kong, Y. Zhao, H. Liu, Y. Yuan, Y. Liu, R. Xia, Y. Wang, M. D. Plumbley, and W. Wang, "Separate anything you describe," *IEEE/ACM TASLP*, vol. 33, pp. 458–471, 2024.

[10] Y. Kwon, D. Lee, D. Kim, and J.-W. Choi, "Self-guided target sound extraction and classification through universal sound separation model and multiple clues," in *Proc. DCASE*, 2025.

[11] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, et al., "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE J-STSP*, vol. 16, no. 6, pp. 1505–1518, 2022.

[12] M. Fazel-Zarandi and W.-N. Hsu, "Cocktail HuBERT: Generalized self-supervised pre-training for mixture and single-source speech," in *Proc. ICASSP*, 2023.

[13] T. Wang, X. Chen, Z. Chen, S. Yu, and W. Zhu, "An adapter based multi-label pre-training for speech separation and enhancement," in *Proc. ICASSP*, 2023.

[14] W. Zhang and Y. Qian, "Weakly-Supervised Speech Pre-training: A Case Study on Target Speech Recognition," in *Proc. Interspeech*, 2023.

[15] Z. Huang, D. Raj, P. García, and S. Khudanpur, "Adapting self-supervised models to multi-talker speech recognition using speaker embeddings," in *Proc. ICASSP*, 2023.

[16] J. Lin, M. Ge, J. Ao, L. Deng, and H. Li, "SA-WavLM: Speaker-aware self-supervised pre-training for mixture speech," in *Proc. Interspeech*, 2024.

[17] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE TASSP*, vol. 27, no. 2, pp. 113–120, 1979.

[18] A. Rouditchenko, Y. Gong, S. Thomas, L. Karlinsky, H. Kuehne, R. Feris, and J. Glass, "Whisper-Flamingo: Integrating visual features into whisper for audio-visual speech recognition and translation," in *Proc. Interspeech*, 2024.

[19] 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.

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

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

[22] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "FiLM: Visual reasoning with a general conditioning layer," in *Proc. AAAI*, vol. 32, 2018.

[23] R. Aihara, Y. Masuyama, F. Paissan, F. G. Germain, G. Wichern, and J. Le Roux, "SUNAC: Source-aware unified neural audio codec," in *Proc. ICASSP*, 2026.

[24] 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.

[25] 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.

[26] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. ICASSP*, 2017.

[27] 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.

[28] 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.

[29] 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.

[30] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proc. ACM MM*, 2014.

[31] P. Warden, "Speech Commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[32] H. Cao, D. G. Cooper, M. K. Keutmann, R. C. Gur, A. Nenkova, and R. Verma, "CREMA-D: Crowd-sourced emotional multimodal actors dataset," *IEEE Trans. Affect. Comput.*, vol. 5, no. 4, pp. 377–390, 2014.

[33] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proc. ICML*, 2017.

[34] J. Turian, J. Shier, G. Tzanetakis, K. McNally, and M. Henry, "One billion audio sounds from GPU-enabled modular synthesis," in *Proc. DAFx*, 2021.

[35] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. ASRU*, 2015.

[36] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, *The MUSDB18 corpus for music separation*, 2017, `https://doi.org/10.5281/zenodo.1117372`.

[37] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *JMLR*, vol. 9, no. 11, 2008.

[38] 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.

[39] 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.

[40] N. Harada, D. Niizumi, Y. Ohishi, D. Takeuchi, and M. Yasuda, "ToyADMOS2025: The evaluation dataset for the DCASE2025T2 first-shot unsupervised anomalous sound detection for machine condition monitoring," in *Proc. DCASE*, 2025.

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

[42] 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.

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

[44] A. Jiang, X. Zheng, B. Han, Y. Qiu, P. Fan, W.-Q. Zhang, C. Lu, and J. Liu, "Adaptive prototype learning for anomalous sound detection with partially known attributes," in *Proc. ICASSP*, 2025.