Adjusting Bias in Anomaly Scores via Variance Minimization for Domain-Generalized Discriminative Anomalous Sound Detection

Masaaki Matsumoto[1], *Takuya Fujimura*[1], *Wen-Chin Huang*[1], *Tomoki Toda*[2]

[1] Graduate School of Informatics, Nagoya University, Nagoya, Japan
[2] Information Technology Center, Nagoya University, Nagoya, Japan

***Abstract*—We propose an anomaly score rescaling method based on variance minimization for domain-generalized anomalous sound detection (ASD). Current state-of-the-art ASD methods face significant challenges due to pronounced domain shifts, which lead to inconsistent anomaly score distributions across domains. One promising existing approach to address this issue is to rescale anomaly scores based on local data density in the embedding space. To enable more flexible and adaptive rescaling, our proposed method introduces weighting parameters into the rescaling process and analytically optimizes them based on the score variance minimization. Experimental evaluations on the DCASE 2021–2024 ASD datasets demonstrate that our proposed method achieves significant improvements on the DCASE 2022–2024 datasets. We also confirm that the proposed method obtains weighting parameters that lead to high ASD performance.**

***Index Terms*—anomalous sound detection, domain generalization, anomaly score rescaling,**

1. INTRODUCTION

Anomaly Sound Detection (ASD) is the task of identifying abnormal sounds from audio data. Since it is difficult to collect anomalous sound data, we need to develop ASD systems by using only normal sound data [1]–[5]. One of the main challenges in the ASD task is domain shift [6]. Domain shifts are variations – in acoustic environments, recording equipment, or operational conditions – that do not affect whether a sample is normal or anomalous. Domain shifts can occur during system deployment, and ASD systems must be able to perform robustly even in domains where only a small amount of data is collected during the development. ASD systems must be robust to domain shifts and perform robustly not only in domains with abundant data but also in domains with only a few samples. Here, it is common to refer to domains with abundant normal data as the *source* domain, and those with only a few normal samples as the *target* domain.

Current state-of-the-art ASD methods predominantly employ discriminative approaches [7]–[11]. These methods leverage labels associated with sounds, such as machine types or operational parameters, and train a feature extractor through the classification task. Anomaly scores are then calculated based on the distance between test samples and the training samples within the discriminative embedding space. The underlying principle is that anomalous sounds are not included in the training data; they are not correctly classified, causing them to deviate from the normal sound distribution in the discriminative embedding space and resulting in high anomaly scores. Although this approach achieves high performance, the limited training data in the target domain still often causes inconsistent anomaly score distributions across domains, as shown in Fig. 1. In such cases, the optimal threshold for distinguishing normal and anomalous samples in the source domain does not generalize well to the target domain.

One promising approach to handle this challenge is the anomaly score rescaling approach [12]. This method rescales anomaly scores based on the local density, where low-density target domains tend to exhibit higher anomaly scores. While this rescaling framework has proven effective, its performance is limited by the assignment of suboptimal fixed hyperparameters across different embedding spaces.

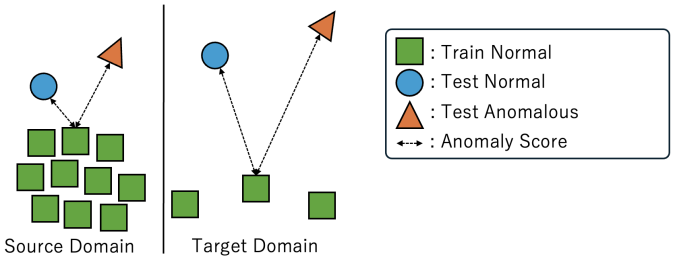

Fig. 1: Discrepancies of anomaly scores between domains.

In this paper, we aim to further enhance domain generalization ability by proposing a new anomaly score rescaling method that automatically adjusts the degree of rescaling according to the embedding space. Our proposed method introduces a weighting parameter into the rescaling process and analytically optimizes it based on the minimization of the variance of anomaly scores over normal samples in both the source and target domains. In experimental evaluations, we demonstrate that our proposed method outperforms existing anomaly score calculation methods on the DCASE 2022–2024 datasets. Furthermore, the experimental analysis shows that our method obtains a weighting parameter that leads to high ASD performance through analytical optimization.

2. ANOMALY SCORE CALCULATION METHODS

2.1. Baseline method

The anomaly score $\mathcal{A}(\boldsymbol{x}, \mathcal{X}_{\text{ref}})$ for a test embedding $\boldsymbol{x}$ is typically calculated as the distance to its nearest neighbor in a reference set $\mathcal{X}_{\text{ref}}$ consiting of normal training samples as follows:

$$\mathcal{A}(\boldsymbol{x}, \mathcal{X}_{\text{ref}}) := \min_{\boldsymbol{y} \in \mathcal{X}_{\text{ref}}} \mathcal{D}(\boldsymbol{x}, \boldsymbol{y}), \tag{1}$$

$$\mathcal{D}(\boldsymbol{x}, \boldsymbol{y}) := \frac{1}{2}(1 - \langle \boldsymbol{x}, \boldsymbol{y} \rangle) \tag{2}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the normalized embeddings with $\|\boldsymbol{x}\| = \|\boldsymbol{y}\| = 1$, $\mathcal{D}(\cdot, \cdot)$ is the cosine distance, $\langle \cdot, \cdot \rangle$ is the inner product operation. We set this approach as our baseline.

2.2. Over- and under-sampling techniques for data imbalance

To address the data imbalance between the source and target domains, techniques such as K-means clustering and SMOTE (Synthetic Minority Over-sampling Technique) are widely employed in the anomaly score calculation process [7], [13]–[15]. K-means clustering is applied to the source domain samples [7], and the obtained centroids and the original target domain samples are used as the reference samples $\mathcal{X}_{\text{ref}}$ in Eq. 1. SMOTE generates synthetic samples in the target domain by linearly interpolating original samples [16]. The augmented target domain samples and the original source domain samples are used as the reference samples $\mathcal{X}_{\text{ref}}$ [13]. These methods aim to mitigate the discrepancies in anomaly score distributions between the source and target domains by balancing the number of samples across domains.

2.3. Anomaly score rescaling

Recently, to address discrepancies in anomaly score distributions, a new approach has been proposed [12]. This method rescales anomaly scores based on the local density of reference samples in the embedding space. It is motivated by the observation that the target domain tends to exhibit higher anomaly scores due to a lack of sufficient reference samples, as shown in Fig. 1.

This method calculates the anomaly score $\mathcal{A}_{\text{scaled}}(\boldsymbol{x}, \mathcal{X}_{\text{ref}})$ as follows:

$$\mathcal{A}_{\text{scaled}}(\boldsymbol{x}, \mathcal{X}_{\text{ref}}) := \min_{\boldsymbol{y} \in \mathcal{X}_{\text{ref}}} \frac{\mathcal{D}(\boldsymbol{x}, \boldsymbol{y})}{b(\boldsymbol{y}, \mathcal{X}_{\text{ref}}, K)}, \tag{3}$$

$$b(\boldsymbol{y}, \mathcal{X}_{\text{ref}}, K) = \frac{1}{K} \sum_{k=1}^{K} \mathcal{D}(\boldsymbol{y}, \boldsymbol{y}_k), \tag{4}$$

where $\boldsymbol{y}_k$ denotes the k-th closest sample to $\boldsymbol{y}$ in $\mathcal{X}_{\text{ref}}$. This method rescale the distance between a test sample $\boldsymbol{x}$ and a reference sample $\boldsymbol{y}$ by dividing it by the local density term $b(\boldsymbol{y}, \mathcal{X}_{\text{ref}}, K)$, which is calculated as the average distance from the reference sample $\boldsymbol{y}$ to its K nearest neighbors within $\mathcal{X}_{\text{ref}}$. By using this local density term, it can prevent the high anomaly scores due to the scarcity of reference samples, thus reducing the discrepancies in anomaly score distributions. Furthermore, unlike techniques such as K-means clustering and SMOTE, this method does not require domain labels and can handle minor domain shifts within the source domain that are not reflected in the domain labels. Despite these advantages, its performance still depends on the manually selected hyperparameter K. Although $K = 16$ was found to perform well across several datasets in their experiments, this fixed value is not optimal for every embedding space.

3. PROPOSED METHOD

To address the performance limitations caused by the suboptimal fixed hyperparameter in the previous anomaly score rescaling method, we propose a new method that adaptively rescales anomaly scores according to each embedding space. Our proposed method calculates the anomaly score $\mathcal{A}_{\text{prop}}(\boldsymbol{x}, \mathcal{X}_{\text{ref}}, \alpha)$ as follows:

$$\mathcal{A}_{\text{prop}}(\boldsymbol{x}, \mathcal{X}_{\text{ref}}, \alpha) := \min_{\boldsymbol{y} \in \mathcal{X}_{\text{ref}}} \left(\mathcal{D}(\boldsymbol{x}, \boldsymbol{y}) - \alpha \cdot b(\boldsymbol{y}, \mathcal{X}_{\text{ref}}, K) \right), \tag{5}$$

where the anomaly score is rescaled by subtracting a bias term, $b(\boldsymbol{y}, \mathcal{X}_{\text{ref}}, K)$, weighted by a newly introduced parameter α. The optimal weighting parameter $\alpha^\star$ is obtained as follows:

$$\boldsymbol{y}^\star(\boldsymbol{z}, \mathcal{X}_{\text{ref}}) = \arg\min_{\boldsymbol{y} \in \mathcal{X}_{\text{ref}}} \mathcal{D}(\boldsymbol{z}, \boldsymbol{y}), \tag{6}$$

$$\alpha^\star = \arg\min_{\alpha} \tag{7}$$

$$\begin{aligned} &\text{Var}(\mathcal{D}(\boldsymbol{z}, \boldsymbol{y}^\star(\boldsymbol{z}, \mathcal{X}_{\text{ref}})) - \alpha \cdot b(\boldsymbol{y}^\star(\boldsymbol{z}, \mathcal{X}_{\text{ref}}), \mathcal{X}_{\text{ref}}, K) \mid \boldsymbol{z} \in \mathcal{X}_{\text{val}}), \\ &= \frac{\text{Cov}(\mathcal{D}(\boldsymbol{z}, \boldsymbol{y}^\star(\boldsymbol{z}, \mathcal{X}_{\text{ref}})), b(\boldsymbol{y}^\star(\boldsymbol{z}, \mathcal{X}_{\text{ref}}), \mathcal{X}_{\text{ref}}, K) | \boldsymbol{z} \in \mathcal{X}_{\text{val}})}{\text{Var}(b(\boldsymbol{y}^\star(\boldsymbol{z}, \mathcal{X}_{\text{ref}}), \mathcal{X}_{\text{ref}}, K) | \boldsymbol{z} \in \mathcal{X}_{\text{val}})}, \end{aligned} \tag{8}$$

where $\text{Var}(\cdot)$ and $\text{Cov}(\cdot)$ are variance and covariance, respectively. $\mathcal{D}(\boldsymbol{z}, \boldsymbol{y}^\star(\boldsymbol{z}, \mathcal{X}_{\text{ref}}))$ is identical to $\mathcal{A}(\boldsymbol{z}, \mathcal{X}_{\text{ref}})$, and Eq. 7 yields the value of α that minimizes the variance of the anomaly scores in the validation set $\mathcal{X}_{\text{val}}$. This variance minimization encourages the alignment of anomaly scores across domains and reduces the need for careful tuning of the hyperparameter K.

For the validation data $\mathcal{X}_{\text{val}}$, we propose four different approaches.

1) *TrainAll:* We use all available normal training data from both source and target domains as $\mathcal{X}_{\text{val}}$.
2) *TrainRandom:* To reduce imbalances between domains in $\mathcal{X}_{\text{val}}$, we construct a balanced validation set consisting of all target training samples and randomly sampled source training samples, such that the number of source samples matches that of the target domain.
3) *TrainCluster:* Another approach to reduce imbalances between domains in $\mathcal{X}_{\text{val}}$ is to use clustering. We first perform K-means clustering on the source domain, and then select the original source samples closest to the centroids as $\mathcal{X}_{\text{val}}$, instead of using the centroids directly. This approach eliminates randomness, unlike *TrainRandom* approach.
4) *TestAll:* We also consider the case where we can utilize test data as $\mathcal{X}_{\text{val}}$. Since test samples observed during the operational phase lack both domain and normal/anomalous labels, we simply use all test as $\mathcal{X}_{\text{val}}$. Assuming that ASD systems are deployed in both the source and target domains, this approach enables the use of balanced validation data including sufficient samples; however, it may include anomalous samples.

Note that *TrainRandom* and *TrainCluster* require the domain labels while *TrainAll* and *TestAll* does not. Since $\mathcal{X}_{\text{ref}}$ includes all training samples and overlaps with $\mathcal{X}_{\text{val}}$, we ensure that $\boldsymbol{y}^\star(\boldsymbol{z}, \mathcal{X}_{\text{ref}}) \neq \boldsymbol{z}$ by excluding $\boldsymbol{z}$ from $\mathcal{X}_{\text{ref}}$.

4. EXPERIMENTAL EVALUATIONS

4.1. Experimental setups

We conducted experimental evaluations using the DCASE 2021–2024 Task 2 Challenge datasets [2]–[5]. These datasets provide labels for machine types, sections, domains, and attributes. The section identifies each individual instance of the same machine type, while the attribute labels reflect the operational state of the machine. The DCASE 2021 and 2022 datasets consist of seven machine types, each with six sections. The DCASE 2023 and 2024 datasets consist of 14 and 16 machine types, respectively, with each machine type having only one section. One section contains approximately 1,000 normal training samples and 400 or 200 test samples. In the training dataset, three samples in DCASE 2021 and ten in each of the DCASE 2022–2024 datasets are from the target domain, while the remaining samples are from the source domain. In the test dataset, the source and target domains, as well as normal and anomalous samples, are approximately balanced (i.e., approximately 100 or 50 samples for each combination of domain and normal/anomalous classes). Each recording is approximately ten seconds long, consisting of a single-channel signal sampled at 16 kHz. Each of these datasets is divided into *dev* and *eval* subsets based on section or machine type, and the evaluation results are aggregated accordingly.

The discriminative feature extractor was similar to that used in [8]. This extractor received an amplitude spectrum and an amplitude spectrogram as input features and processed them in parallel using two separate neural networks, a spectrum network and a spectrogram network. The spectrum network consisted of 1D convolutional layers, while the spectrogram network consisted of 2D convolutional layer-based ResNet [17] blocks, Squeeze-and-Excitation [18] blocks, and multilayer perceptrons. The final output was obtained by concatenating the outputs from spectrum and spectrogram networks. For the spectrogram, we used a DFT size of 1024 and a hop length of 512. The frequency range was restricted to 200-8000 Hz. We trained the feature extractor for 16 epochs jointly using machine type, section, domain, and attribute labels. The optimizer was AdamW [19] with a fixed learning rate of 0.001 and a batch size was set to 64. For the loss function, we used Sub-cluster AdaCos (SCAC) [20] with the number of sub-clusters set to 16 and a fixed scale parameter. While previous work [7] showed that fixed class centers lead to better performance

Table 1: Mean (± Standard Deviation) of official score [%] for DCASE 2021–2024 datasets using fixed SCAC loss. † requires the domain labels.

Method	2021		2022		2023		2024	
	dev	eval	dev	eval	dev	eval	dev	eval
Baseline	67.31 ± 0.51	66.28 ± 0.65	68.97 ± 0.75	63.79 ± 0.92	62.11 ± 0.98	57.00 ± 1.51	58.63 ± 0.71	51.59 ± 0.73
w/ K-means clustering†	65.27 ± 0.59	64.02 ± 0.59	70.49 ± 0.81	61.82 ± 1.04	**63.63 ± 0.70**	59.17 ± 1.44	59.24 ± 0.96	52.15 ± 0.63
w/ SMOTE†	**67.48 ± 0.48**	**66.43 ± 0.62**	69.46 ± 0.74	64.63 ± 0.95	63.24 ± 0.75	59.27 ± 1.52	59.36 ± 0.78	52.16 ± 0.73
Rescaling ($K = 8$)								
Previous [12]	63.38 ± 1.39	61.43 ± 1.65	64.08 ± 1.81	62.68 ± 0.77	58.94 ± 1.34	65.32 ± 1.37	**60.08 ± 1.63**	51.47 ± 1.38
Prop (*TrainAll*)	52.46 ± 2.11	49.83 ± 2.53	69.10 ± 1.55	66.34 ± 0.83	61.20 ± 1.64	**67.41 ± 1.58**	58.56 ± 1.74	54.10 ± 1.25
Prop (*TrainRandom*)†	60.69 ± 3.61	59.15 ± 2.93	70.60 ± 0.96	66.84 ± 1.05	62.62 ± 2.16	66.18 ± 1.52	58.20 ± 2.59	54.06 ± 1.35
Prop (*TrainCluster*)†	60.44 ± 3.60	58.55 ± 2.87	**70.63 ± 1.00**	66.84 ± 1.02	62.23 ± 1.93	66.25 ± 1.80	57.92 ± 2.84	**54.40 ± 1.06**
Prop (*TestAll*)	64.02 ± 1.04	62.75 ± 1.24	70.47 ± 1.19	65.28 ± 0.67	**64.97 ± 0.95**	**67.91 ± 0.96**	59.43 ± 1.85	**56.58 ± 0.68**
Rescaling ($K = 16$)								
Previous [12]	63.62 ± 1.38	61.58 ± 1.68	63.33 ± 1.68	61.90 ± 0.85	58.71 ± 1.58	63.92 ± 1.47	59.67 ± 1.71	48.18 ± 0.98
Prop (*TrainAll*)	51.07 ± 3.17	50.46 ± 1.99	66.71 ± 1.92	65.85 ± 0.87	60.40 ± 1.99	66.09 ± 1.38	58.18 ± 1.81	54.16 ± 1.27
Prop (*TrainRandom*)†	60.65 ± 3.32	58.72 ± 3.17	70.32 ± 1.11	**67.59 ± 0.92**	62.81 ± 2.07	65.63 ± 1.75	58.47 ± 2.02	53.52 ± 1.11
Prop (*TrainCluster*)†	60.21 ± 3.45	58.04 ± 3.07	70.20 ± 1.14	67.55 ± 0.87	62.41 ± 1.90	65.46 ± 1.81	57.96 ± 2.37	53.63 ± 1.11
Prop (*TestAll*)	63.93 ± 1.10	62.47 ± 1.31	69.43 ± 2.77	65.24 ± 0.92	**65.43 ± 0.99**	67.39 ± 0.86	59.35 ± 1.68	**56.21 ± 0.68**

Table 2: Mean (± Standard Deviation) of official score [%] for DCASE 2021–2024 datasets using trainable SCAC loss. † requires the domain labels.

Method	2021		2022		2023		2024	
	dev	eval	dev	eval	dev	eval	dev	eval
Baseline	68.82 ± 0.60	65.22 ± 0.64	70.82 ± 0.63	67.32 ± 0.56	63.96 ± 1.10	63.68 ± 3.49	60.42 ± 1.03	56.16 ± 0.71
w/ K-means clustering†	67.01 ± 0.98	63.96 ± 0.62	70.23 ± 1.11	64.34 ± 0.70	65.53 ± 0.99	65.47 ± 2.83	61.54 ± 1.16	55.71 ± 0.64
w/ SMOTE†	69.05 ± 0.57	**65.30 ± 0.55**	**71.12 ± 0.67**	67.86 ± 0.60	65.31 ± 0.93	65.60 ± 2.55	61.82 ± 0.99	56.20 ± 0.78
Rescaling ($K = 8$)								
Previous [12]	69.39 ± 0.85	64.47 ± 0.78	66.53 ± 1.39	66.60 ± 0.80	61.87 ± 1.66	67.23 ± 0.89	62.74 ± 1.14	52.59 ± 0.93
Prop (*TrainAll*)	66.99 ± 0.94	62.26 ± 0.75	67.79 ± 1.26	67.61 ± 0.70	64.94 ± 1.37	**69.75 ± 1.22**	62.63 ± 1.15	55.02 ± 0.70
Prop (*TrainRandom*)†	68.52 ± 1.26	64.50 ± 0.80	70.98 ± 0.88	68.80 ± 0.62	**66.51 ± 0.84**	69.04 ± 1.38	**62.74 ± 0.70**	**57.05 ± 0.77**
Prop (*TrainCluster*)†	67.72 ± 0.87	63.60 ± 0.36	70.87 ± 0.78	68.89 ± 0.61	66.44 ± 0.94	69.33 ± 1.53	62.44 ± 1.02	56.22 ± 0.89
Prop (*TestAll*)	68.75 ± 0.79	64.62 ± 0.50	**71.66 ± 0.63**	67.36 ± 0.57	66.19 ± 0.69	**69.91 ± 1.25**	62.40 ± 1.48	**58.63 ± 0.79**
Rescaling ($K = 16$)								
Previous [12]	**69.47 ± 0.83**	64.48 ± 0.83	65.41 ± 1.40	65.77 ± 0.64	60.95 ± 1.33	66.28 ± 0.54	62.39 ± 1.12	49.82 ± 0.84
Prop (*TrainAll*)	66.33 ± 0.99	61.83 ± 0.70	67.93 ± 1.11	67.69 ± 0.83	64.22 ± 1.48	69.19 ± 1.32	62.59 ± 1.10	55.64 ± 1.22
Prop (*TrainRandom*)†	68.51 ± 1.15	64.59 ± 0.80	71.04 ± 0.81	**69.03 ± 0.62**	66.46 ± 0.87	68.61 ± 1.50	62.72 ± 0.74	56.76 ± 1.03
Prop (*TrainCluster*)†	67.76 ± 0.87	63.73 ± 0.41	70.85 ± 0.74	68.97 ± 0.55	66.24 ± 0.85	69.07 ± 1.42	62.28 ± 0.89	56.20 ± 1.21
Prop (*TestAll*)	68.69 ± 0.76	64.61 ± 0.56	**71.79 ± 0.68**	67.80 ± 0.57	66.23 ± 0.71	69.07 ± 1.01	62.36 ± 1.39	**57.88 ± 0.70**

than trainable class centers, we found that trainable class centers performed better. Accordingly, we conducted experiments using both fixed and trainable centers in the SCAC loss. Additionally, we applied Mixup [21] to the input waveforms with a probability of 0.5.

We evaluated our proposed method with four comparison previous backend methods: the baseline method (Eq.1), the baseline method with K-means clustering or SMOTE described in Sec.2.2, and the anomaly score rescaling method [12] described in Sec. 2.3. For the baseline method with K-means clustering, we set the number of clusters to 16. For the SMOTE, we set the oversampling ratio to 5% and the number of neighbors to 5. For the anomaly score rescaling approach, we set K to 8 and 16 following the previous work [12]. For *TrainCluster* in our proposed method, we used K-means clustering with the number of clusters set to match the number of samples in the target domain (i.e., three for DCASE 2021 and ten for DCASE 2022–2024).

As evaluation metrics, we used the official DCASE metrics for each dataset: the harmonic mean of the area under the receiver operating characteristic (ROC) curve (AUC) and the partial AUC (pAUC) with $p = 0.1$ over all machine types and domains. We calculated the arithmetic mean and standard deviation of the official scores across ten independent trials.

4.2. Experimental results

Tables 1 and 2 show the evaluation results when using fixed and trainable class centers for the SCAC loss, respectively. For the DCASE 2022–2024 datasets, our proposed method consistently achieves high performance, significantly improving performance in several subsets. For example, in 2023 eval of Table 1, baseline, K-means clustering, SMOTE, and the previous rescaling with $K = 8$ achieved 57.00%, 59.17%, 59.27%, and 65.32%, respectively, while our proposed *TrainAll* achieved 67.41% with $K = 8$. Additionally, *TrainAll* of our proposed method achieves high performance without requiring domain labels, whereas the K-means clustering and SMOTE techniques rely on them. Comparing Tables 1 and 2, we can see that using trainable SCAC loss improves overall performance. Although this reduces the relative performance gain of our method, it still consistently contributes to performance improvement.

We can also see that the previous rescaling method [12] can cause performance degradation, whereas our proposed method improves or keeps the baseline performance in most cases of the DCASE 2022–2024 datasets, regardless of whether the hyperparameter K is set to 8 or 16. For example, in 2023 dev of Table 2, the previous method with $K = 8$ degrades performance from 63.96% of baseline to 61.87%, whereas the proposed method *TrainAll* with $K = 8$ achieves 64.94%. In contrast, in 2023 eval of Table 2, the previous method

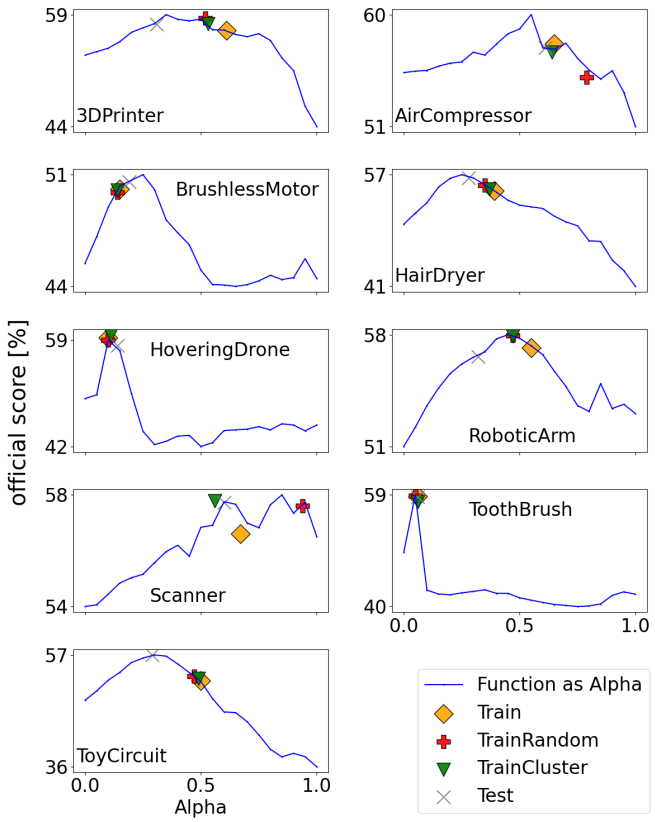

Fig. 2: Graphs of the official score improvement for DCASE 2024 evaluation data (one trial with a fixed SCAC loss, $K = 16$). The horizontal axis shows α, and the vertical axis shows official score [%]. Plotted points indicate the α selected by each validation data selection method (Orange: *TrainAll*, Red: *TrainRandom*, Green: *TrainCluster*, Gray: *TestAll*) for each machine.

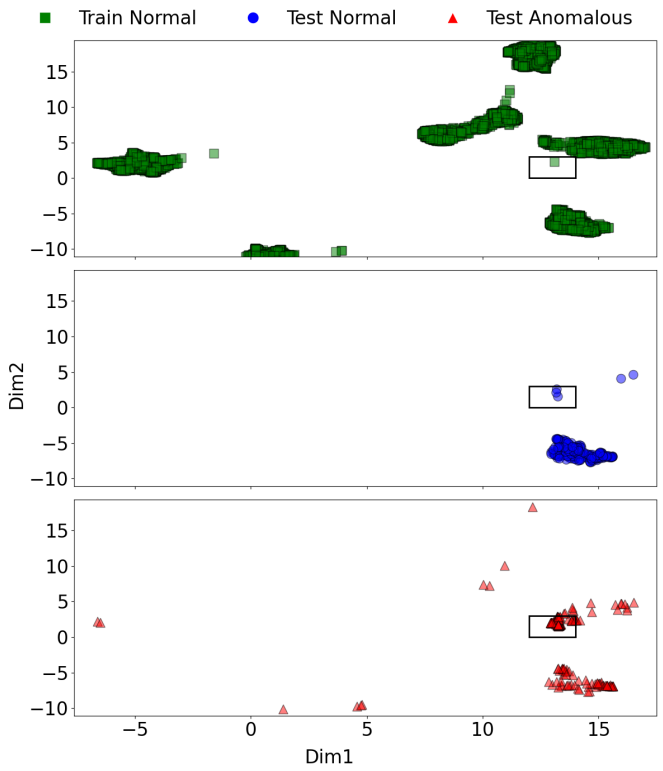

Fig. 3: The plots show the embedding of the test samples of section 3 and training samples of all sections for the *pump* machine from the DCASE 2021 dataset (one trial with a fixed SCAC loss). The AUC for these test samples of the baseline method, the previous rescaling using $K = 8$, and the proposed method using $K = 8$ and *TrainAll* were 86.69%, 67.01%, and 43.98%, respectively.

with $K = 8$ improves the performance from 63.68% of baseline to 67.23%, where the proposed method *TrainAll* with $K = 8$ also improves the performance to 69.75%.

Regarding validation data selection, the comparison among *TrainAll*, *TrainCluster*, and *TrainRandom* shows that the best choice varies depending on the dataset, and there is no consistent trend. On the other hand, *Test* approach achieves the highest performance in most cases. For example, in 2023 dev of Table 1, the proposed method *Test* with $K = 16$ achieves 65.43%, in 2024 eval of Table 1, the proposed method *Test* with $K = 8$ achieves 56.58%, and in 2024 eval of Table 2, the proposed method *Test* with $K = 8$ achieves 58.63%. This suggests the importance of using validation datasets in which the source and target domains are balanced with sufficient samples.

We confirm that $\alpha^\star$ indeed leads to high performance. Fig. 2 illustrates the relationship between α and the official score, along with the $\alpha^\star$ obtained using each validation data selection method. This figure is generated for each machine type in the DCASE 2024 evaluation subset, when using the fixed SCAC loss and $K = 16$ under a specific random seed. Here, the proposed method with $\alpha = 0$ is equivalent to the baseline method. The figure clearly shows that the optimal value of α varies across machine types, and that the proposed method adaptively selects values of α that yield high performance for each type.

Despite the overall performance improvements in the DCASE 2022–2024 datasets, we observe that the proposed methods and the previous rescaling method [12] exhibit degraded performance on the DCASE 2021 dataset compared to other methods. To investigate the reason for this degradation, we examine the embedding space. Figure 3 visualizes a partial excerpt of the embedding space for *pump* machine using UMAP [22]. In the highlighted frame within the figure, we can observe that a single training sample is embedded apart from the other training samples, while many anomalous samples are located nearby. Due to the low density of training samples in that area, the rescaling method inadvertently reduces the anomaly scores of anomalous sounds, leading to performance degradation.

5. CONCLUSION

This paper introduced an anomaly score rescaling method based on variance minimization for domain-generalized ASD. Our proposed method introduced weighting parameters into the local data density based rescaling process and analytically optimized them based on the score variance minimization. Experimental results demonstrated that (1) the proposed method significantly improves performance over existing anomaly score calculation methods; (2) ore stable with respect to random seed than the previous rescaling method [12]; and (3) the weighting parameters derived through our variance minimization scheme adaptively rescale anomaly scores for each machine type, leading to high performance.

6. ACKNOWLEDGMENT

This work is partly supported by JST AIP Acceleration Research JPMJCR25U5.

REFERENCES

[1] Y. Koizumi, Y. Kawaguchi, K. Imoto, *et al.*, "Description and discussion on DCASE2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring," in *Proc. DCASE*, 2020, pp. 81–85.

[2] Y. Kawaguchi, K. Imoto, Y. Koizumi, *et al.*, "Description and discussion on DCASE 2021 challenge task 2: Unsupervised anomalous detection for machine condition monitoring under domain shifted conditions," in *Proc. DCASE*, 2021, pp. 186–190.

[3] K. Dohi, K. Imoto, N. Harada, *et al.*, "Description and discussion on DCASE 2022 challenge task 2: Unsupervised anomalous sound detection for machine condition monitoring applying domain generalization techniques," in *Proc. DCASE*, 2022, pp. 1–5.

[4] K. Dohi, K. Imoto, N. Harada, *et al.*, "Description and discussion on DCASE 2023 challenge task 2: First-shot unsupervised anomalous sound detection for machine condition monitoring," in *DCASE*, 2023, pp. 31–35.

[5] T. Nishida, N. Harada, D. Niizumi, *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, pp. 111–115.

[6] K. Wilkinghoff, T. Fujimura, K. Imoto, J. L. Roux, Z.-H. Tan, and T. Toda, "Handling domain shifts for anomalous sound detection: A review of DCASE-related work," *arXiv preprint arXiv:2503.10435*, 2025.

[7] K. Wilkinghoff, "Design choices for learning embeddings from auxiliary tasks for domain generalization in anomalous sound detection," in *Proc. ICASSP*, 2023, pp. 1–5.

[8] K. Wilkinghoff, "Self-supervised learning for anomalous sound detection," in *Proc. ICASSP*, 2024, pp. 276–280.

[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, pp. 1–5.

[10] X. Zheng, A. Jiang, B. Han, *et al.*, "Improving anomalous sound detection via low-rank adaptation fine-tuning of pre-trained audio models," in *Proc. SLT*, 2024, pp. 969–974.

[11] A. Jiang, B. Han, Z. Lv, *et al.*, "Anopatch: Towards better consistency in machine anomalous sound detection," in *Proc. Interspeech*, 2024, pp. 107–111.

[12] K. Wilkinghoff, H. Yang, J. Ebbers, F. G. Germain, G. Wichern, and J. Le Roux, "Keeping the balance: Anomaly score calculation for domain generalization," in *Proc. ICASSP*, 2025, pp. 1–5.

[13] A. Jiang, X. Zheng, B. Han, *et al.*, "Adaptive prototype learning for anomalous sound detection with partially known attributes," in *Proc. ICASSP*, 2025, pp. 1–5.

[14] Z. Lv, A. Jiang, B. Han, *et al.*, "AITHU system for first-shot unsupervised anomalous sound detection," DCASE2024 Challenge, Tech. Rep., 2024.

[15] A. Jiang, X. Zheng, Y. Qiu, *et al.*, "Thuee system for first-shot unsupervised anomalous sound detection," DCASE2024 Challenge, Tech. Rep., 2024.

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

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[18] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.

[19] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. ICLR*, 2019.

[20] K. Wilkinghoff, "Sub-cluster adacos: Learning representations for anomalous sound detection," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–8.

[21] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *Proc. ICLR*, 2018.

[22] L. McInnes, J. Healy, N. Saul, and L. Grossberger, "UMAP: Uniform Manifold Approximation and Projection," *The Journal of Open Source Software*, vol. 3, no. 29, 2018, 63 pages.