ANOMALOUS SOUND DETECTION WITH ENSEMBLE OF AUTOENCODER AND BINARY CLASSIFICATION APPROACHES

Technical Report

Ibuki Kuroyanagi[1,2]*, Tomoki Hayashi*[1,2]*, Yusuke Adachi*[1,2]*,*
Takenori Yoshimura[1]*, Kazuya Takeda*[2]*, Tomoki Toda*[2]

[1] Human Dataware Lab. Co., Ltd., Nagoya, Japan
tomoki.hayashi@hdwlab.co.jp
[2] Nagoya University, School of Informatics, Nagoya, Japan
kuroyanagi.ibuki@g.sp.m.is.nagoya-u.ac.jp

ABSTRACT

This paper describes a solution with the ensemble of two unsupervised anomalous sound detection (ASD) methods for the DCASE2021 Challenge Task 2. The first ASD method is based on a sequence-level autoencoder with section ID regression and a self-attention architecture. We introduce the data augmentation techniques such as SpecAugment to boost up the performance and combine the simple scorer module for each section and each domain to address the domain shift problem. The second ASD method is based on a binary classification model using metric learning, which utilizing task-irrelevant outliers as pseudo-anomalous data and considering the centroid of normal and outlier data in the feature space. As a countermeasure against the domain shift problem, we perform data augmentation using Mixup with data from the target domain, resulting in a stable performance for each section. On the development set, our method achieves a harmonic mean of 76.59% harmonically averaged over of area under the curve (AUC) and partial AUC ($p = 0.1$) of all machines, sections, and domains.

***Index Terms*—** Anomalous sound detection, autoencoder, binary classification, metric learning

1. INTRODUCTION

In this paper, we describe our solution for DCASE2021 Challenge Task 2 [1] using two unsupervised ASD approaches. The first approach is a sequence-level autoencoder using ID regression [2] and a self-attention architecture [3, 4]. Unlike standard frame-wise autoencoders, the proposed autoencoder receives the entire sequence of input features and reconstructs it at once by utilizing the self-attention architecture. Furthermore, we explicitly utilize section ID information as the inputs to avoid the confusion between anomalous and normal sounds in different IDs.

The second approach is based on a binary classification model using metric learning, which utilizes pseudo-anomalous data. This approach assumes that carefully selected task-irrelevant outlier data can be substituted as anomalous data [5], which makes it possible to train the model under the classification problem that discriminates between anomalous and normal even without the real anomalous data. To further enhance this approach, we introduce a novel metric learning method that considers the class centroids of the model's feature space [6].

The main difference from DCASE2020 Challenge Task2 [7] is the existence of the domain shift. There are two types of domain: source and target. The source domain data accounts for the majority of the training data while the target domain data only exists in tiny amounts. It is necessary to develop a model that can detect anomalous sound in both domains with only the unbalanced training data. To tackle with this problem, we use different techniques for each approach. In the autoencoder approach, we build the scorer module with a Gaussian distribution for each section and each domain, which calculates the likelihood of reconstruction errors. This allows us to absorb the difference in the reconstruction error range caused by the domain shift. In the binary classification approach, we perform the fine-tuning with data augmentation. We fine-tune the model with the target domain data and pseudo-target domain data generated by Mixup [8] to adapt it to the target domain for the model trained only on the source domain.

Experimental evaluation with DCASE2021 Task 2 dataset [1] demonstrates that 1) both of our approaches significantly outperform the baselines systems, 2) the autoencoder approach and binary classification approach have different specialty, depending on the target machine type, and 3) the ensemble of both approach achieves the best performance, resulting in a harmonic mean of area under the curve (AUC) and partial AUC ($p = 0.1$) of 81.62% in the source domain and 69.47% in the target domain.

2. METHOD

2.1. Auto-encoder approach

The first approach is a sequence-level auto-encoder with ID regression based on our previous work [2]. The overview is shown in Fig. 1. The input features are log Mel-spectrogram extracted from audio and one-dimensional section ID determined from the filename. The auto-encoder model consists of Conformer blocks [4], accepting the sequence of features, and calculating the reconstruction error. Instead of using the reconstruction error as the score, we construct scorer modules using the reconstruction errors. We use a Gaussian distribution with full covariance for the score calculation to model the distribution of the reconstruction errors using the subset of training data, which is not used for the training of the auto-encoder. Finally, we use the likelihood of the reconstruction errors as a frame-wise anomaly score and then perform post-processing to squeeze the time dimension. For more detail, see our previous

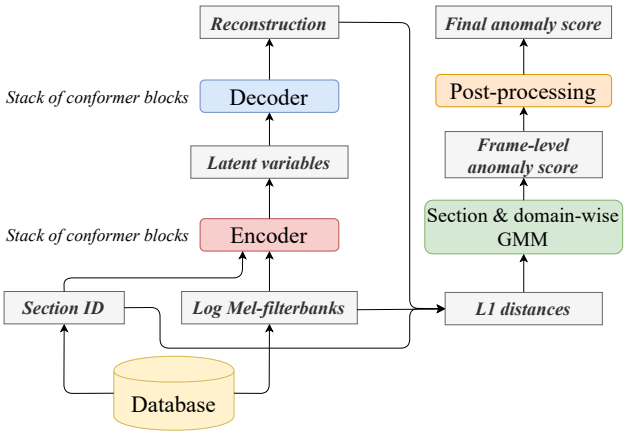

Figure 1: *Overview of the proposed autoencoder approach.*

work [2].

To boost the auto-encoder approach's performance, we introduce SpecAugment (time masking and frequency masking) [9] and dropout for the input feature sequence. Inspired by the interpolation deep neural network (IDNN) [10], we apply SpecAugment and dropout for the input feature sequence not only in training but also in the inference. During the inference, we replicate the input sequence and apply different masking for each of them. Then, we calculate each reconstruction error and integrate them using a pooling operation (e.g., average or max). This method enables us to obtain the gain like the model ensemble even with the single model.

In previous work [2], we have constructed a single scorer module for each machine; however, to address the domain shift problem, we build them for each section and each domain (e.g., source and target). This approach is simple but effective, enabling us to absorb the difference of the anomaly score range between the sections and domains.

2.2. Binary classification approach

The second approach is a binary classification model using metric learning. The overview is shown in Fig. 2. To train a binary classification model, we use the normal data of the target section of the machine as positive examples, and the other normal data including the different machines as negative examples. A waveform of each example is normalized over each machine type and then converted to log Mel-spectrogram with 1024 FFT points, 256 shift points, and 256 dimensional Mel basis. We use the PyTorch Image

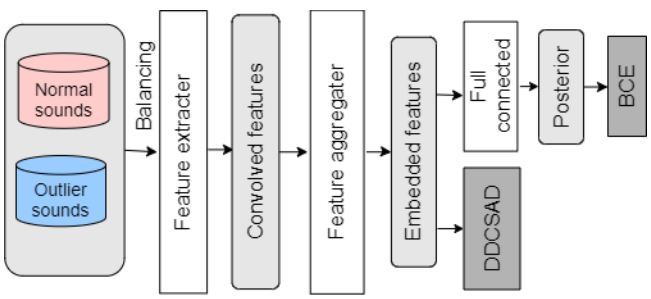

Figure 2: *Overview of the proposed binary classification approach.*

Models [11] as a feature extraction module in the binary classification model. The output features are then averaged in the time and frequency directions to create an embedding vector that was aggregated in the channel direction. We apply deep double-centroids semi-supervised anomaly detection (DDCSAD) loss to this embedding vector, which enables us to minimize the within-class variance and maximize the between-class variance in the feature space [6]. The embedding vector is further trained to minimize a binary cross-entropy loss by passing through dropout, ReLU, and full-connected layer. We build the model for the source domain and then perform fine-tuning for the target domain. When training the source domain, only the data from the source domain is used. On the other hand, when fine-tuning for the target domain, we make pseudo target domain data by Mixup with source and target domain data [8]. The use of Mixup increases the variation of each class and created data with an intermediate representation between positive and negative examples, resulting in a more accurate classification model. For inference, we combine the posterior probability with the normalized distance between the embedding vector and the centroid of the normal class to calculate the anomaly score [6].

To make a variation for the model ensemble, we change the pseudo-anomalous example selection, introduce additional data augmentation such as Gaussian Noise and Volume control, change the architecture of the feature extraction part, and use an additional loss function, ArcFace [12]. For negative example selection, we change to select samples within the same dataset (e.g., MIMII [13] and ToyADMOS2 [14]), resulting in more stable performance among the sections. ArcFace is a loss function to achieve a clear geometric interpretation in the feature space, and the combination of ArcFace with DDCSAD brings further improvements. Since the binary classification approach build the model for each section, the performance is less stable than autoencoder approach. Therefore, we use multiple models including ResNet34 [15], ResNeXt50 [16], and EfficientNet b3 [17]) in PyTorch Image Models to stabilize the performance.

2.3. Model Ensemble

To further improve the performance, we conduct the model ensemble. We select the N-best model on the developed data for each machine and each domain and then integrate the outputs of these models to obtain the final score. Before ensembling the scores for each model, we normalize them to be mean zero and variance one. Finally, we combine the normalized scores using the following four methods: *average*, *median*, *maximum*, and *ranking*. We select the best method for each machine and each section.

In the case of the autoencoder approach, we select the N-best models and ensemble them for each machine and each domain. In the case of the binary classification approach, we select the N-best models of each section and then ensemble them for each machine and each domain, i.e., we ensemble $N \times S$ models for each machine and each domain, where S represents the number of sections in the validation set. The number of N is optimized for each machine and each domain.

3. EXPERIMENTAL EVALUATION

3.1. Experimental condition

We conducted an experimental evaluation using the DCASE2021 Challenge Task 2 dataset [13, 14]. The dataset consisted of the

Table 1: *Evaluation results. The values represent the harmonic mean of AUC [%] and pAUC ($p = 0.1$) [%] for each section of each domain. The value in the column "all / harmean" represents the harmonic mean of AUC and pAUC over all machines, sections, and domains.*

	ToyCar		ToyTrain		fan		gearbox		pump		silder		valve		all
Method	source	target	source	target	source	target	source	target	source	target	source	target	source	target	harmean
Baseline (AE)	59.44	54.74	64.31	51.99	59.14	56.72	56.42	61.04	63.85	53.01	67.09	55.71	52.43	51.45	57.29
Baseline (MobileNetV2)	57.19	55.89	58.81	50.77	63.31	61.58	65.54	60.72	62.20	57.36	65.43	52.17	53.99	55.17	59.39
AE	80.41	63.05	80.50	61.28	71.82	65.97	62.69	70.01	72.61	62.41	86.01	62.01	80.60	64.30	69.05
BC	57.91	58.68	76.23	49.04	67.36	59.36	74.85	74.59	72.12	59.86	80.64	57.24	86.18	70.10	69.08
AE ensemble	**83.29**	68.70	**81.06**	**62.83**	74.37	69.75	64.14	72.32	74.60	**65.68**	86.12	65.41	82.94	67.81	71.67
BC ensemble	60.93	63.55	75.16	55.40	**82.29**	66.62	**74.49**	70.58	75.20	60.70	**89.28**	57.69	92.70	79.35	73.29
AE+BC ensemble (mix)	79.90	**70.08**	80.23	59.85	82.25	**71.58**	72.95	**76.25**	**77.29**	64.03	89.06	**68.49**	**93.03**	**80.15**	**76.59**
AE+BC ensemble (max)	**83.29**	68.70	**81.06**	**62.83**	**82.29**	69.75	**74.49**	72.32	75.20	**65.68**	**89.28**	65.41	92.70	79.35	75.68

normal/anomalous operating sounds of seven types of toy/real machines: *ToyCar*, *ToyTrain*, *fan*, *gearbox*, *pump*, *slider*, and *valve*. Each machine included six sections, and each section was divided into two domains, i.e., source and target. Each recording was a single-channel, approximately 10-sec length audio sampled at 16 kHz. The number of training data was from around 1,000 samples in the source domain and only 3 samples in the target domain, and that of development data was from around 200 samples in both domains, which depending on the machine type. The training data included only normal sounds, but the development data included both normal and anomalous sounds to check the anomaly detection performance. To verify the performance, we compared the following models:

Baseline (AE): The autoencoder-based official baseline [1]. It was trained on the normal training data by minimizing the reconstruction error in the sense of mean squared error.

Baseline (MobileNetV2): The classification-based official baseline [1]. It was trained under classification problem of section ID to minimize the cross entropy.

AE: The proposed sequence-level autoencoder model. The model was trained for 50,000 steps using Adam optimizer [18] with Warmup scheduler [3]. The batch size was set to 64. The number of warmup steps was 8,000. The hyperparameters were optimized for each machine and each domain, including Mel-spectrogram extraction condition (e.g., shift size and Mel basis), model architecture (e.g., the number of blocks, units, and kernel size), and post-processing. In SpecAugment, the number of time masks was set to 50, and the width range was from one to five. The number of frequency masks was five, and the width range was from zero to ten. The dropout rate for the input sequence was set to 0.2. During the inference, we replicated the input sequence ten times and then integrated them with average, max, or median pooling.

BC: The proposed binary classification-based method. The model was ResNet34. The size of spectrogram was 256×256. The model was trained for 8,000 steps using Adam optimizer, the learning rate for fully-connected layer 1.0e-3, the learning rate for convolution layer 5.0e-4, OneCycleLR scheduler [19], and the batch size 64. The ratio of normal data and outlier data was set to 1:1 in the mini-batch. When fine-tuning the target domain, we trained the pre-trained model created by the source domain for 800 steps. However, sampling should be done at this time so that the mini-batch always contains 16 samples of target domain data or pseudo-target domain data obtained by mixing up the target domain and source domain.

AE ensemble: The ensemble of the proposed autoencoder models. The value of N was selected from $\{3, 5, 10, 20\}$ and the ensemble method were optimized for each machine and each domain.

BC ensemble: The ensemble of the proposed binary-classification models. The value of N was set to two and the ensemble method was *average*.

AE+BE ensemble (mix): The ensemble of **AE ensemble** and **BC ensemble**. We ensembled two ensembled scores with *average*; in other words, this can be seen as the weighted averaged score of the autoencoder and binary classification approaches.

AE+BE ensemble (max): The ensemble of **AE ensemble** and **BC ensemble**. We took the maximum value between **AE ensemble** and **BC ensemble** for each machine and each domain.

The hyperparameters of each model and the post-processing parameters were optimized for each section and each domain. The evaluation metric was the harmonic mean of an area under the curve (AUC) and a partial AUC (pAUC) ($p = 0.1$).

3.2. Experimental results

The experimental results are shown in Table 1. From the comparison among the different approach, the autoencoder-based approach outperformed the binary classification approach in the machine type *ToyCar* and *ToyTrain*, while the binary classification approach outperformed the autoencoder-based approach in the machine type *gearbox* and *valve*. The result showed that each approach focused on different features. Furthermore, we were able to further improve the performance by ensembling the models within the same method. In particular, the binary classification approach has a problem of creating a model for each machine for each section, which tends to increase the variability of the score. However, by ensembling the models, we were able to reduce the variability of the score, which greatly contributed to the improvement of the overall score. Finally, we ensembled the two approaches. The ensemble allowed us to obtain a higher score by complementing both the excellent results of the two approaches and outperform the baseline significantly.

4. CONCLUSION

This paper presented an ensemble approach combining sequence-level autoencoder and binary classification model using metric learning. Experimental evaluation showed that the proposed approach significantly outperformed baseline systems. By ensembling

our completely different methods, we were able to obtain higher scores. In particular, note that there are several machine types for which the score of **AE+BE ensemble (mix)** is much better than that of **AE+BE ensemble (max)**. We believe that this is because each method focused on different features to detect anomalous sounds. These results suggest that it is important to ensemble models that focus on different features. Future work includes integrating the ensembled models and develop a method to obtain the same or better performance with fewer models.

5. ACKNOWLEDGMENT

This paper was partly supported by a project, JPNP20006, commissioned by NEDO.

6. REFERENCES

[1] Y. Kawaguchi, K. Imoto, Y. Koizumi, N. Harada, D. Niizumi, K. Dohi, R. Tanabe, H. Purohit, and T. Endo, "Description and discussion on DCASE2021 Challenge Task 2: Unsupervised anomalous sound detection for machine condition monitoring under domain shifted conditions," *arXiv preprint arXiv:2106.04492*, 2021.

[2] T. Hayashi, T. Yoshimura, and Y. Adachi, "Conformer-based ID-aware autoencoder for unsupervised anomalous sound detection," DCASE2020 Challenge, Tech. Rep., 2020.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of 2017 Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[4] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, *et al.*, "Conformer: Convolution-augmented Transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[5] P. Primus, V. Haunschmid, P. Praher, and G. Widmer, "Anomalous sound detection as a simple binary classification problem with careful selection of proxy outlier examples," *arXiv preprint arXiv:2011.02949*, 2020.

[6] I. Kuroyanagi, T. Hayashi, K. Takeda, and T. Toda, "Anomalous sound detection using a binary classification model and class centroids," *arXiv preprint arXiv:2106.06151*, 2021.

[7] Y. Koizumi, Y. Kawaguchi, K. Imoto, T. Nakamura, Y. Nikaido, R. Tanabe, H. Purohit, K. Suefusa, T. Endo, M. Yasuda, and N. Harada, "Description and discussion on DCASE2020 Challenge Task 2: Unsupervised anomalous sound detection for machine condition monitoring," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, 2020, pp. 81–85.

[8] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[9] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proceedings of Interspeech 2019*, 2019, pp. 2613–2617.

[10] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi, "Anomalous sound detection based on interpolation deep neural network," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 271–275.

[11] R. Wightman, "PyTorch Image Models," https://github.com/rwightman/pytorch-image-models, 2019.

[12] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

[13] R. Tanabe, H. Purohit, K. Dohi, T. Endo, Y. Nikaido, T. Nakamura, and Y. Kawaguchi, "MIMII DUE: Sound dataset for malfunctioning industrial machine investigation and inspection with domain shifts due to changes in operational and environmental conditions," *arXiv preprint arXiv:2006.05822*, 2021.

[14] 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," *arXiv preprint arXiv:2106.02369*, 2021.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[16] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[17] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 687–10 698.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of 3rd International Conference on Learning Representations, ICLR 2015*, 2015, pp. 1–15.

[19] L. N. Smith and N. Topin, "Super-Convergence: Very fast training of neural networks using large learning rates," *arXiv preprint arXiv:1708.07120*, 2017.