ENSEMBLE OF COMPLEMENTARY ANOMALY DETECTORS UNDER DOMAIN SHIFTED CONDITIONS

Jose A. Lopez, Georg Stemmer, Paulo Lopez-Meyer, Pradyumna S. Singh
Juan A. del Hoyo Ontiveros, Hector A. Courdourier

Intel Corporation
{jose.a.lopez, georg.stemmer, paulo.lopez.meyer, pradyumna.s.singh}@intel.com
{juan.del.hoyo.ontiveros, hector.a.cordourier.maruri}@intel.com

ABSTRACT

We present our submission to the DCASE2021 Challenge Task 2, which aims to promote research in anomalous sound detection. We found that blending the predictions of various anomaly detectors, rather than relying on well-known domain adaptation techniques alone, gave us the best performance under domain shifted conditions. Our submission is composed of two self-supervised classifier models, a probabilistic model we call NF-CDEE, and an ensemble of the three – the latter obtained the top rank in the DCASE2021 Challenge Task 2.

Index Terms— DCASE, anomaly detection, domain shift, machine condition monitoring, machine health monitoring.

1. INTRODUCTION

The DCASE2021 Challenge Task 2 is concerned with identifying anomalous behavior from a target machine using sound recordings [1]. A major difference between this task and other DCASE tasks is that it is not supervised. Accordingly, the available training data only contains samples from the normal-state distributions. A further complication added to this challenge is that the acoustic characteristics of the training data and of the test data are different – this condition is known as domain shift and there are some known results for reducing the performance gap between the training and test data [2, 3, 4, 5, 6, 7, 8]. In our experiments, while we recognize the potential of these techniques, we did not generally gain much from using these methods alone.

In our submission, we used two self-supervised classifiers that classified the section IDs similar to the approach several teams followed in DCASE2020 [9, 10, 11, 12, 13]. For a third model, we introduce a model that relies on several normalizing flows to estimate the conditional density of input Mel spectrogram sections and use their combined outputs to produce an anomaly score [14, 15, 16, 17, 18, 19, 20, 21, 22].

In the sequel we describe each model, how it was trained, its hyperparameters, and their respective results. In order to put the results into perspective, we include the scores for the baseline autoencoder and MobileNetV2 models on Tables 1 and 2, respectively. The data used in this challenge is 16 KHz, single-channel, audio. For more details, please see [1, 23, 24].

	ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
h-mean AUC	0.6249	0.6171	0.6324	0.6597	0.6192	0.6674	0.5341
h-mean pAUC	0.5236	0.5381	0.5338	0.5276	0.5441	0.5594	0.5054

Table 1: Baseline Autoencoder Scores

	ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
h-mean AUC	0.5604	0.5746	0.6156	0.6670	0.6189	0.5926	0.5651
h-mean pAUC	0.5637	0.5161	0.6302	0.5916	0.5737	0.5600	0.5264

Table 2: Baseline MobileNetV2 Scores

2. ARCHITECTURES

The first model described below builds on the work from [9]. In particular, the encoder network has been updated to use 1D convolutions rather than 2D. The input to this model is a spectrogram with or without a Mel transformation. The second model builds on the well-known WaveNet architecture [25] by adding an x-vector [26] classification head after the dilated convolutions – in a sense, the WaveNet functions as a time-series encoder for the x-vector component. Both models mentioned above are trained to reduce the cross entropy loss between predictions and the section IDs. The third model differs from the first two models in that it is completely unsupervised and attempts to learn several distributions of some Mel spectrogram bins conditioned on the remaining bins. We also describe a fourth model, a 1D convolutional autoencoder, which we did not include in our submission but we believe may be of interest to the community. We call these approaches complementary because of the different input modalities and learning approaches. The last system described is an ensemble of the first three models described above.

All our development was done using PyTorch [27] and spectrograms were computed using nnAudio [28]. The third model additionally used the Pyro [29] probabilistic programming library.

2.1. XVector1D

A high-level view of the architecture of the first model is shown in Figure 1. We denote additive margin softmax as AMS [30].

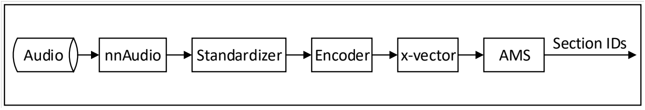

Figure 1: XVector1D High-level Architectures

In Figure 1, we use the term “standardizer” as a preprocessing step done before passing data to the rest of the network. In the simplest case, it is a batch-norm layer with the learnable parameters disabled. In this way, this batch-norm will perform the usual

ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
STFT	MEL	STFT	MEL	STFT	STFT	STFT
AutoDIAL	batch-norm	AutoDIAL	AutoDIAL	AutoDIAL	AutoDIAL	AutoDIAL
C(128,192)	C(128,192)	C(128,192)	C(128,192)	C(128,192)	C(128,192)	C(128,192)
5 x C(192,192)	5 x C(192,192)	5 x C(192,192)	4 x C(192,192)	5 x C(192,192)	5 x C(192,192)	5 x C(192,192)
			C(192,90)			

Table 3: Input And Encoder Parameters

frequency-wise normalization once the running statistics have converged. However, in other cases, “standardizer” can mean an AutoDIAL layer which mixes the statistics from the source and target domains for normalization [4]. In early experiments we evaluated the more general domain adaptation technique from [6]; however, we found the performance similar to AutoDIAL but, in our implementation, much more computationally expensive.

The encoder used in this model includes 1D convolutions with kernel size 3 and leaky-relu activations. The number of layers varied with machine type as shown on Table 3 – in this table and going forward, we use “C” to mean a 1D convolution.

The x-vector component used here remains largely the same as in [9] except the interface to the encoder was adapted (as expected) to accept the 1D encoder output.

2.1.1. Preprocessing

This model did not use any special preprocessing or augmentation. The logarithm was taken for both the STFT and the Mel spectrograms. All spectrograms were computed with frequency min and max values set to 100 and 8000 Hertz, respectively. Mel spectrograms were computed using 128 bins.

2.1.2. Training & Results

The model was trained to predict the section ID meta-data parameter using the cross entropy loss function. We found that the spectrogram parameters had a big effect on the performance. Parameters like the number of input samples, the number of points used for the FFT, the hop length can have a significant effect. We generally used the AdamW optimizer with the default learning rate of 1×10^{-3} and weight decay set to 1×10^{-4}. However, we used ASGD with the default learning rate (and no weight decay) for gearbox. Generally, the training losses converge more slowly using ASGD but sometimes the slower trajectory spends more epochs close to an optimal region with respect to AUC and this can yield better results[1]. During training, random contiguous audio clips were sampled and the spectrograms were computed on the fly using nnAudio [28]. The training was usually run for 300 epochs, using all the training data from the development and evaluation datasets. Lastly, we computed the average embedding, during training, using the embedding from the layer prior to the final AMS classification layer. At test time, the average embedding was used to compute the cosine and Mahalanobis distances to the test embedding which served as additional options for anomaly scores. Table 4 shows the results, and, Table 5 shows the effect on performance when using AutoDIAL.

2.2. WaveNet-XVector

We explored the use of a WaveNet model processing the audio samples directly. For details on the architecture we refer the reader to the original publication [25]. In the original paper the authors explain that the model can be readily adapted to classification tasks and in their classification experiment they add a mean pooling layer after the dilated convolutions followed by “a few non-causal convolutions”. The training proceeds with two loss terms: one for predicting the next sample and the other is the classification loss. We follow this procedure in that we use a mean pooling layer (with kernel size 10) and train with the two loss functions but instead of using a few convolutions, we use an x-vector component, with AMS top layer, as with the XVector1D model. In this way, one can consider this model a variant of the XVector1D model which uses an audio-only encoder. For the WaveNet encoder, we used a single block with 14 layers. Gearbox and pump used 64 channels for the dilation, residual, and skip channels. The other machines used 32 channels. For valve we used an AutoDIAL standarizer, and the rest used a batch-norm.

	ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
batch size	128	64	128	64	128	128	64
input samples	16384	16384	16384	98000	16384	16384	98000
no. Mels	2048	128	2048	128	2048	2048	2048
no. FFT	4096	1024	4096	1024	4096	4096	4096
hop	80	512	512	80	512	512	512
scoring	cosine	mahalanobis	softmax	mahalanobis	softmax	softmax	softmax
h-mean AUC	0.6702	0.7193	0.7171	0.8342	0.7799	0.7871	0.9032
h-mean pAUC	0.6233	0.6772	0.7295	0.7443	0.6684	0.6728	0.7724

Table 4: XVector1D Scoring Results

	ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
AutoDIAL	2.79%	-12.90%	4.49%	23.41%	2.24%	1.31%	0.46%

Table 5: Relative Change In XVector1D Score Using AutoDIAL

2.2.1. Preprocessing

For valve and ToyTrain we used the Teager-Kaiser energy operator (TKEO) to preprocess the audio [31, 32, 33, 34]. The motivation was that, because the valve noises are sparse and impulsive events, the noise suppression provided by the Teager-Kaiser operator would improve the signal-to-noise (SNR) ratio in the valve recordings. Despite improving the results for valve and ToyTrain, the improvement was modest.

2.2.2. Training & Results

To train this model, we used the Adamax optimizer with the default learning rate for 200 epochs in the same manner as XVector1D, with 16384 input samples. Table 6 shows the performance of this model using softmax scoring. Table 7 shows the effects on performance due to AutoDIAL and TKEO independently.

	ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
batch size	128	128	128	64	64	128	128
h-mean AUC	0.5843	0.6641	0.8122	0.7156	0.7543	0.7184	0.7297
h-mean pAUC	0.5629	0.5696	0.8025	0.5964	0.6506	0.6239	0.6206

Table 6: WaveNet-XVector Scoring Results

2.3. NF-CDEE

For our third system, we began by attempting to model the probability density function of the Mel spectrograms of the machine sounds, for a single machine, using normalizing flows. We used the Pyro [29] probabilistic programming library to develop this model. We

[1] We used the harmonic mean of AUC and pAUC harmonic means to assign a single score to a model configuration. For gearbox, the experiment using ASGD had an 8.89% greater score.

	ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
AutoDIAL	-0.54%	-0.11%	-3.13%	-2.62%	-1.71%	-2.22%	4.97%
TKEO	-3.15%	7.69%	-27.04%	-5.98%	-9.70%	-14.08%	3.30%

Table 7: Relative Change On WaveNet-XVector Score Using AutoDIAL & TKEO

found that training a model to fit a distribution with the same dimensions as Mel bins to be somewhat unstable. In order to improve the stability we instead estimated several conditional densities and trained them in a single model, minimizing the sum of their negative log-likelihoods. We consider this model an ensemble of conditional density anomaly detectors. Hence, we call this model NF-CDEE, because it uses normalizing flows and it is a conditional density estimator ensemble. Each conditional density estimator fits the distribution of a n-bin segment of input spectrograms conditioned on the remaining bins. This reduces the instability due to dimensionality. The parameter n and the amount of overlap are tunable by the user. For this work, we chose $n = 32$ with no overlap. Each normalizing flow uses a single conditional spline with 16 count-bins and the default hidden layer dimensions – these are also tunable but in our experiments they did not significantly affect the performance.

To summarize, each estimator outputs the probability $p(s_A|s_{A^c})$ where s is a vector of dimension equal to the number of Mel bins m that is indexed by the set $\mathcal{I} = \{1, \ldots, m\}$. A is an n-element subset of $\mathcal{I}$, and A^c is its complement $\mathcal{I} - A$. We define the likelihood of the normal state as:

$$p(\text{normal}) = \prod_i p(s_{A_i}|s_{A_i^c}) \tag{1}$$

where $i \in [1, \ldots, k]$ and k is a positive integer provided by the user – it is the number of estimators in the ensemble. Here, we used $A_1 = \{1, \ldots, 32\}$, $A_2 = \{33, \ldots, 64\}$, and so forth. To train the model, we minimize the negative logarithm of $p(\text{normal})$. Therefore, the output of NF-CDEE is the sum of the individual negative log-likelihoods.

2.3.1. *Training & Results*

To train this model we converted the input audio to 256-bin Mel spectrograms, computed using 8192-point FFTs with hop-length 512, and applied frequency-wise normalization before passing to the conditional density estimators. Unlike the self-supervised models, the spectrograms were pre-computed and spectrogram windows were passed to the network in the same manner as [35]. Each model was trained with all the sections of the development (or evaluation) training data, per machine type – except for fan for which we trained a model for each section. To further reduce training instability, caused by the normalizing flow determinant computation, we take the mean across the time dimension. This last step was important for stabilizing the training of the ensemble. As previously stated the loss function used was the sum of the negative log-likelihoods and this also served as the anomaly score. Figure 2 shows the inference process.

For the optimizer, we used the same optimizer as the XVector1D, with gradient clipping. In our experiments this model generally needs to train for about 50 epochs. Table 8 shows the results, sampling 192 spectrogram frames in batches of 32.

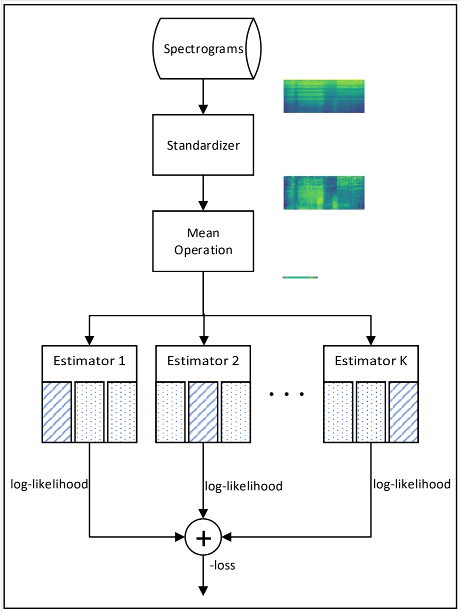

Figure 2: NF-CDEE Inference

	ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
h-mean AUC	0.8657	0.7797	0.7866	0.8081	0.6993	0.7483	0.6130
h-mean pAUC	0.7831	0.6031	0.6024	0.6513	0.5655	0.6054	0.5275

Table 8: NF-CDEE Scoring Results

2.4. 1D CNN Autoencoder

The model described here is a 1D convolutional autoencoder that reconstructs (Mel) spectrograms. We excluded this model for several reasons including that, like NF-CDEE, its performance was strongest for ToyCar but NF-CDEE was also strong for other machines. Additionally, WaveNet-XVector offered stronger fan performance than either XVector1D or NF-CDEE and resulted in a better ensemble when including three models.

The architecture of this model is shown in Table 9. The bottleneck for this autoencoder was inspired by [36] in that the time dimension was mostly preserved[2]. In our post-DCASE2020 analyses, we found that preserving the time dimension to be a key factor for the success of the autoencoder in [36]. Additionally, we found the scoring methods in [36] to be effective at improving AUC performance in (spectrogram) autoencoders. For example, the scoring methods in [36] can improve the results from [35]. Table 9 shows the architecture of this model, which uses leaky-relu activations, kernel size 3, and a batch-norm standardizer.

ToyCar	ToyTrain	fan	gearbox	pump	slider	valve
C(128,256)	C(128,192)	C(128,256)	C(128,256)	C(128,192)	C(128,192)	C(128,192)
3 x C(256,256)	4 x C(192,192)	2 x C(256,256)	5 x C(256,256)	3 x C(192,192)	5 x C(192,192)	3 x C(192,192)
C(256,20)	C(192,30)	C(256,10)	C(256,30)	C(192,90)	C(192,90)	C(192,90)
C(20,256)	C(30,192)	C(10,256)	C(30,256)	C(90,192)	C(90,192)	C(90,192)
3 x C(256,256)	4 x C(192,192)	2 x C(256,256)	5 x C(256,256)	3 x C(192,192)	5 x C(192,192)	3 x C(192,192)
C(256, 128)	C(192, 128)	C(256, 128)	C(256, 128))	C(192, 128)	C(192, 128)	C(192, 128)

Table 9: 1D CNN Autoencoder

[2] In [36] the time dimension was not reduced at all because causal convolutions were used.

2.4.1. Training & Results

The training for this model followed the same approach as NF-CDEE, using spectrogram windows as in [35]. The loss function used was mean absolute error (MAE). However, for gearbox, fan, and ToyCar we attached an x-vector classifier head to the bottleneck layer and additionally included a cross-entropy loss term for the meta-data. Moreover, for gearbox, fan, and ToyCar we employed a special weighting scheme during training and at test time.

The intuition behind the weighting is that if one knew the SNR of a frequency bin, one could weigh the reconstruction loss using this information, giving greater importance to bins with greater SNR. Estimating the SNR is not straightforward, so we used the frequency bin variance in its place. The weight vector w is precomputed with elements given by $w_f = \frac{1}{\sigma_f^2}$, where σ_f^2 denotes the variance of the fth Mel bin[3], using the training data for each domain and each section.

Table 10 shows the results sampling 192 frames, from 128-bin Mel spectrograms, in batches of 64. The spectrogram hop length was set to 512. The E1 and E2 scoring methods referenced in Table 10 come from [36] and are repeated here for the reader's convenience.

$$E1(X, \hat{X}) = \frac{1}{FT} \sum_{f=1}^{F} \left[\sum_{t=1}^{T} \left(X_{f,t} - \hat{X}_{f,t} \right) \right]^2 \tag{2}$$

and

$$E2(X, \hat{X}) = \frac{1}{FT} \sum_{f=1}^{F} \left| \sum_{t=1}^{T} \left(X_{f,t} - \hat{X}_{f,t} \right) \right| \tag{3}$$

where $X \in \mathbb{R}^{F \times T}$ and $\hat{X} \in \mathbb{R}^{F \times T}$ are the true and reconstructed spectrograms. F and T are natural numbers that denote the frequency and time dimensions, respectively.

	ToyCar	**ToyTrain**	**fan**	**gearbox**	**pump**	**slider**	**valve**
scoring	E2	MAE	E1	E1	E1	E2	E2
no. FFT	8192	2048	8192	4096	2048	8192	4096
h-mean AUC	0.8663	0.7180	0.7265	0.7287	0.6981	0.7022	0.6117
h-mean pAUC	0.7502	0.6023	0.5738	0.5992	0.5951	0.5782	0.5230

Table 10: 1D CNN Autoencoder Scoring Results

2.5. Ensemble

For the last system we combined the first three models (described in Sections 2.1, 2.2, and 2.3) by first standardizing the training data scores and then searching over a grid of convex combinations, similar to [37].

We could have included the autoencoder, for example, by ensembling separately with each system but we did not have time to explore this or other ensembling alternatives. As it stands, this model influenced the development of XVector1D, particularly its encoder, and the selection of hyperparameters for NF-CDEE. Table 11 shows the results of the ensemble of the first three models.

	ToyCar	**ToyTrain**	**fan**	**gearbox**	**pump**	**slider**	**valve**
WaveNet weight	0.03	0.03	1	0.04	0.32	0.02	0
XVector1D weight	0.06	0.55	0	0.61	0.68	0.52	1
NF-CDEE weight	0.91	0.42	0	0.35	0	0.46	0
h-mean AUC	0.8745	0.7756	0.8122	0.8613	0.7958	0.8287	0.9032
h-mean pAUC	0.7837	0.7048	0.8025	0.7635	0.6790	0.6925	0.7724

Table 11: Ensemble Scoring Results

3. CONCLUSIONS

We have outlined our submission to the DCASE2021 Challenge Task 2, which featured a domain shift between the training and test distributions. We found it concerning that domain adaptation methods that seem to do well for other modalities, especially vision, do not seem to work as well for audio (at least in our implementations). This discrepancy gives the DCASE2021 Challenge a greater relevance, because it highlights the need for the audio community to generate more effective domain adaptation methods for audio.

As the XVector1D, WaveNet-XVector, and NF-CDEE models (respectively) ranked 11, 52, and 31, it is clear that the three were indeed complementary and that ensembling is a good option for improving results under domain shifted testing conditions. We do not find the lower individual ranks too concerning because the scores are for single models, as opposed to ensembles, and because the rankings do not fully reflect the performance on individual machine categories.

Of the models we investigated, we find NF-CDEE to be particularly promising because it performed well and is unsupervised. In real-world settings it is not always practical to leverage meta-data, even when it is possible to do so. Moreover, we expect the ensembling nature of the model to perform better under domain shifted conditions. We plan to develop this model further going forward.

4. 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 dcase 2021 challenge task 2: Unsupervised anomalous sound detection for machine condition monitoring under domain shifted conditions," *arXiv preprint arXiv:2106.04492*, 2021.

[2] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," 2020.

[3] Y. Li, N. Wang, J. Shi, X. Hou, and J. Liu, "Adaptive batch normalization for practical domain adaptation," *Pattern Recognition*, vol. 80, pp. 109–117, 2018.

[4] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò, "Autodial: Automatic domain alignment layers," 2017.

[5] ——, "Just dial: Domain alignment layers for unsupervised domain adaptation," 2017.

[6] M. Mancini, L. Porzi, S. R. Bulò, B. Caputo, and E. Ricci, "Boosting domain adaptation by discovering latent domains," 2018.

[7] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation," 2018.

[8] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," 2015.

[3] The weight vector was also scaled to have a max element of 1.

[9] J. A. Lopez, H. Lu, P. Lopez-Meyer, L. Nachman, G. Stemmer, and J. Huang, "A speaker recognition approach to anomaly detection," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, November 2020, pp. 96–99.

[10] R. Giri, S. V. Tenneti, F. Cheng, K. Helwani, U. Isik, and A. Krishnaswamy, "Self-supervised classification for detecting anomalous sounds," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, November 2020, pp. 46–50.

[11] T. Inoue, P. Vinayavekhin, S. Morikuni, S. Wang, T. Hoang Trong, D. Wood, M. Tatsubori, and R. Tachibana, "Detection of anomalous sounds for machine condition monitoring using classification confidence," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, November 2020, pp. 66–70.

[12] 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," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, November 2020, pp. 170–174.

[13] Q. Zhou, "Arcface based sound mobilenets for dcase 2020 task 2," DCASE2020 Challenge, Tech. Rep., July 2020.

[14] E. G. Tabak and C. V. Turner, "A family of nonparametric density estimation algorithms," *Communications on Pure and Applied Mathematics*, vol. 66, no. 2, pp. 145–164.

[15] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," 2016.

[16] I. Kobyzev, S. Prince, and M. Brubaker, "Normalizing flows: An introduction and review of current methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1–1, 2020.

[17] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," 2021.

[18] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural spline flows," 2019.

[19] H. M. Dolatabadi, S. Erfani, and C. Leckie, "Invertible generative modeling using linear rational splines," 2020.

[20] L. Dinh, D. Krueger, and Y. Bengio, "Nice: Non-linear independent components estimation," 2015.

[21] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," 2017.

[22] D. Ha, A. Dai, and Q. V. Le, "Hypernetworks," 2016.

[23] 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," *In arXiv e-prints: 2006.05822, 1-4*, 2021.

[24] 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.

[25] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," 2016.

[26] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[27] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[28] K. W. Cheuk, H. Anderson, K. Agres, and D. Herremans, "nnaudio: An on-the-fly gpu audio to spectrogram conversion toolbox using 1d convolution neural networks," 2020.

[29] E. Bingham, J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. A. Szerlip, P. Horsfall, and N. D. Goodman, "Pyro: Deep universal probabilistic programming," *J. Mach. Learn. Res.*, vol. 20, pp. 28:1–28:6, 2019.

[30] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Processing Letters*, vol. 25, no. 7, p. 926–930, Jul 2018.

[31] P. Maragos, J. Kaiser, and T. Quatieri, "Energy separation in signal modulations with application to speech analysis," *IEEE Transactions on Signal Processing*, vol. 41, no. 10, pp. 3024–3051, 1993.

[32] P. Maragos, J. F. Kaiser, and T. F. Quatieri, "On amplitude and frequency demodulation using energy operators," *IEEE Transactions on Signal Processing*, vol. 41, no. 4, pp. 1532–1550, Apr. 1993.

[33] A. Georgogiannis and V. Digalakis, "Speech emotion recognition using non-linear teager energy based features in noisy environments," in *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, 2012, pp. 2045–2049.

[34] H. Li, H. Zheng, and L. Tang, "Gear fault detection based on teager-huang transform," *International Journal of Rotating Machinery*, vol. 2010, pp. 1–9, 2010.

[35] A. Ribeiro, L. Matos, P. Pereira, E. Nunes, A. Ferreira, P. Cortez, and A. Pilastri, "Deep dense and convolutional autoencoders for unsupervised anomaly detection in machine condition sounds," DCASE2020 Challenge, Tech. Rep., July 2020.

[36] V. K. Agrawal and S. S. Maurya, "Unsupervised detection of anomalous sounds for machine condition monitoring," DCASE2020 Challenge, Tech. Rep., July 2020.

[37] P. Daniluk, M. Gozdziewski, S. Kapka, and M. Kosmider, "Ensemble of auto-encoder based systems for anomaly detection," DCASE2020 Challenge, Tech. Rep., July 2020.