UNSUPERVISED DETECTION OF ANOMALOUS SOUNDS USING ABNORMAL SOUND SIMULATION ALGORITHM AND AUTO-ENCODER CLASSIFIER

Technical Report

Haifeng Wen [1], *Shaoyang Shi*[2], *Chuang Shi*[3]

University of Electronic Science and Technology of China, Chengdu, China
[1] wenhaifeng@std.uestc.edu.cn
[2] 2018011211016@std.uestc.edu.cn
[3] shichuang@uestc.edu.cn

ABSTRACT

This report described our contribution to Unsupervised Detection of Anomalous Sounds on DCASE 2020 challenge (Task2). In our work, we made some changes to the algorithm for simulating abnormal sound referred to the idea of abnormal sound simulation. Besides, to make use of the simulated abnormal sound, the change of output of the classification system based on Auto-encoder and binary cross entropy used for system's training were done. The experiment results show a significant improvement performance comparing with baseline system's results. In this report, we propose two systems with the above basic ideas, which are based on fully connected neural networks and convolutional neural networks (CNNs), respectively.

Index Terms— Unsupervised learning, Anomalous sound detection, Abnormal sound simulation, DNN, GMM, Autoencoder.

1. INTRODUCTION

Anomaly detection in sound (ADS) has been paid much attention nowadays due to the abnormal sounds' damage to performance of machines. In detail, supervised detection in sounds and unsupervised detection in sounds are two kinds of ADS. However, it's almost impossible to get all the information of abnormal sounds in reality because of actual anomalous sounds' high variability, which means that supervised-ADS has no much useful application to do for us. For unsupervised-ADS, the only allowed dataset for train was normal sounds, which leads to most of the traditional detection system hard to increase the true positive rate (TPR) and to decrease the false positive rate (FPR). However, TPR and FPR are the important performance measures of ADS. It is necessary to increase TPR and decrease FPR simultaneously to improve the overall performance of unsupervised-ADS. Therefore, to get high TPR under low FPR, anomalous sound simulation algorithm is a good method.

Algorithm 1: Simulation algorithm of abnormal sound in latent vector space

1: **Input:** Generator $\boldsymbol{G}$, $\boldsymbol{zv}$ and $\boldsymbol{zu}$, n and l
2: Fit $\boldsymbol{zv}$ and $\boldsymbol{zu}$ to get gmm and $gmm2$ with $mixture.GaussianMixture$
3: Calculate $-gmm2.score\_samples(zu)$, and sort it from large to small
4: Take the $0.1 * batchsize$-th one as ϕ_z
5: **While** n<l do
6: generator a sample z with $gmm.sample()$
7: calculate the score of z by $-gmm2.score\_sample(z)$
8: **if** the score > ϕ_z
9: n=n++
10: $\boldsymbol{G}$(z) to get $\boldsymbol{xa}$
11: **end if**
12: **end while**
13: **Output:** $\boldsymbol{xa}$,and the number of $\boldsymbol{xa}$ is n

Figure 1: Abnormal sound simulation algorithm

In our study, we still used the auto-encoder (AE) as the classification system. Before the normal sounds data from the Development dataset were used as input of the AE, we used the offered external data resources from IDMT-ISA-ELECTRIC-ENGINE as the input of the variational AE (VAE) which is used as the abnormal sound simulation system. With the abnormal sound simulation algorithm shown in Figure 1, the outputs of VAE can be regarded as the simulated abnormal sounds [3]. Then, by using the simulated abnormal sounds and normal sounds data from the Development dataset as the input of AE, we can get the mean square error (MSE) as the final output.

The idea of simulating abnormal sounds with external data resources and the Auto-encoder Classifier are used to highlight the clustering and outlying of normal data, in other words, to increase TPR and decrease FPR simultaneously. Through the test, the results showed a significant improvement performance comparing with baseline system's results. The rest of this report will introduce the pretreatment, classification system and show the results of our systems.

2. PRETREATMENT

In this section we introduce the log mel spectrum and the abnormal sound simulation system.

2.1. Log Mel Spectrum

Log mel spectrum is one of the classical acoustic features because it can simulate the process of human ear's perception of the external sound. In our work, the log mel spectrum is also used as the acoustic feature.

2.2. Abnormal Sound Simulation System

Based on VAE, Figure 2 shows the structure of the abnormal sound simulation system [3]. In this system, the external data from IDMT-ISA-ELECTRIC-ENGINE is used as the training data. The latent vector was corresponding to normal distribution after the fit of Gaussian mixture model. Then with the abnormal sound

simulation algorithm and generator of VAE, the simulated abnormal sounds were acquired. This section will introduce the structure of abnormal sound simulation.

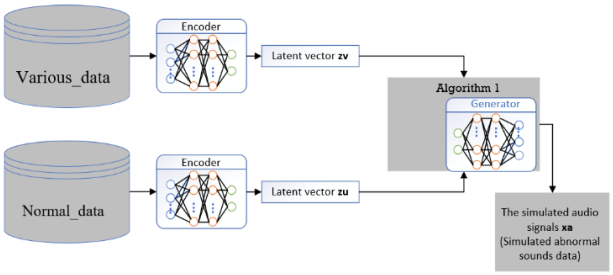

Figure 2: Abnormal sound simulation system [2]

The structure of encoder and generator (just like the decoder of VAE) based on fully connected layers and convolutional layers are shown as table 2.1 and table 2.2, respectively. "Dense (512)-BN-Relu" denotes a dense layer has the 512 units followed by batch normalization and ReLU activation. "Conv1D(filter=64, kernel =5)-BN-ReLU" denotes a CNN1D layer has the 64 filter and a kernel of size 5 followed by batch normalization, ReLU activation and max pooling.

The training procedure of the abnormal sound simulation system is actually the training procedure of VAE, namely, the parameters Θ_E and Θ_G to minimum the error between the reconstructed simulated abnormal audio signal and its corresponding input audio signal. Besides, to increase the speed and decrease the pay for this training procedure, it's necessary to normalize the normal distribution of latent vectors, which is also related to the parameters training. The detailed training procedure is referred to [3].

Table 2.1: The dense-based VAE. The input feature is with the size of $n_{mels} \times frame = 640$.

Layer Name	Settings
Encoder Input	Shape= (640)
Encoder	Dense (512)-BN-ReLU Dense (512)-BN-ReLU Dense (512)-BN-ReLU Dense (512)-BN-ReLU
Encoder Output	z_mean = Dense (32) z_log_var = Dense (32) z = Sampling (z_mean, z_log_var)
Generator Input	Shape= (32)
Generator	Dense (512)-BN-ReLU Dense (512)-BN-ReLU Dense (512)-BN-ReLU Dense (512)-BN-ReLU
Generator Output	Dense (640)

3. CLASSIFICATION SYSTEM

We need a system based on the ideas proposed in Section 1, which can make full use of the simulated abnormal sounds as well. In this section, we propose a system named autoencoder-based classifier with better performance comparing with baseline system. We will introduce the structure and the training method of our system.

Table 2.2: The CNN-based VAE. The input feature is with the size of $(n_{mels} \times frame = 640,1)$.

Layer Name	Settings
Encoder Input	Shape= (640,1)
Encoder	Conv1D(filter=64, kernel =5)-BN-ReLU Maxpooling 2 Conv1D(filter=128, kernel =5)-BN-ReLU Maxpooling 2 Conv1D(filter=256, kernel =5)-BN-ReLU Maxpooling 2 Flatten-Dense(128, Relu)
Encoder Output	z_mean = Dense (16) z_log_var = Dense (16) z = Sampling (z_mean, z_log_var)
Generator Input	Shape= (16)
Generator	Dense (80*256) - Reshape (80,256) Conv1D(filter=256, kernel =5)-BN-ReLU Upsampling 2 Conv1D(filter=128, kernel =5)-BN-ReLU Upsampling 2 Conv1D(filter=64, kernel =5)-BN-ReLU Upsampling 2
Generator Output	Conv1D(filter=1, kernel=5)

3.1. Autoencoder-Based Classifier

In our implementation, the classification system can be implemented by two different structures. The structures are shown in Table 3.1 and Table 3.2.

In our autoencoder classifier, the system consists of an encoder and decoder that can be trained, and the AE is followed by a MSE function and a sigmoid function.

$$error = mse(output_{decoder}, inputs) \quad (1)$$

$$output_{ae\_classifier} = sigmoid(error) \quad (2)$$

Through sigmoid function, we can map the AE output into (0,1). Finally, the Anomaly Score is the same as $output_{ae}$.

Table 3.1: The dense-based autoencoder classifier. The input feature is with the size of $n_{mels} \times frame = 640$.

Layer Name	Settings
Encoder Input	Shape= (640)
Encoder	Dense (512)-BN-ReLU Dense (512)-BN-ReLU Dense (512)-BN-ReLU Dense (512)-BN-ReLU
Encoder Output	z = Dense (32)
Decoder Input	Shape= (32)
Decoder	Dense (512)-BN-ReLU Dense (512)-BN-ReLU Dense (512)-BN-ReLU Dense (512)-BN-ReLU
Decoder Output	Dense (640)
Lambda	Error=MSE (Decoder-output, input)

Output	Output=Sigmoid (Error)

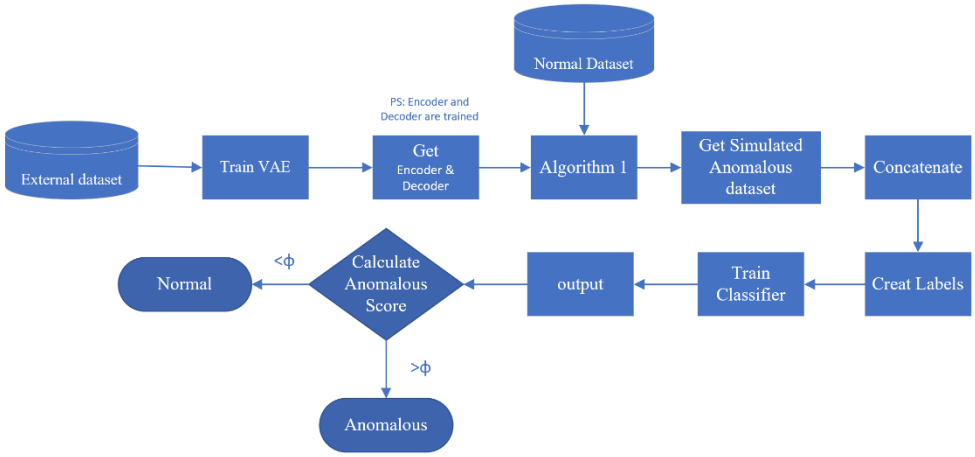

Figure 3: Training process

Table 3.2: The CNN-based autoencoder classifier. The input feature is with the size of ($n_{mels} \times frame = 640,1$).

Layer Name	Settings
Encoder Input	Shape= (640,1)
Encoder	Conv1D(filter=64, kernel =5)-BN-ReLU Maxpooling 2 Conv1D(filter=128, kernel =5)-BN-ReLU Maxpooling 2 Conv1D(filter=256, kernel =5)-BN-ReLU Maxpooling 2 Flatten-Dense(128, Relu)
Encoder Output	z = Dense (16)
Decoder Input	Shape= (16)
Decoder	Dense (80*256) - Reshape (80,256) Conv1D(filter=256, kernel =5)-BN-ReLU Upsampling 2 Conv1D(filter=128, kernel =5)-BN-ReLU Upsampling 2 Conv1D(filter=64, kernel =5)-BN-ReLU Upsampling 2
Decoder Output	Conv1D(filter=1, kernel=5)
Lambda	Error=MSE (Flatten(Decoder-output), Flatten(input))
Output	Output=Sigmoid (Error)

3.2. Classifier Training Method

We create a list of labels, labeling normal audio data and simulated anomalous audio data, where 0 and 1 represent normal he anomalous, respectively. We named the list of labels as y_{true} which is the expected output of the classification system.

Because the output of our classification system is $sigmoid(x)$, and after we build a label list, we can train AE similarly to train a binary classifier. Obviously, we need to choose binary cross entropy binary_crossentropy as the loss function:

$$L(y, \hat{y}) = -\frac{1}{N}\sum_{i=0}^{N}(y * \log(\hat{y}_i) + (1-y) * \log(1-\hat{y}_i)). \quad (3)$$

The $binary\_crossentropy$ measures how far each category's prediction is from the true value (0 or 1), and then averages these category errors to obtain the final loss.

The output of out AE has the following characteristics:

1. When the input is normal data, the anomalous score will be very small, i.e. $mse(outputs, inputs)$ will be small.
2. When the input is anomalous data, the anomalous score will be large, i.e. $mse(outputs, inputs)$ will be large.

Generally, it is difficult for us to ensure that ordinary autoencoders have the above-mentioned features, however, we can use the mapping and $binary\_crossentropy$ as a loss function to train the network to have the above-mentioned features.

4. EXPERIMENTS

The whole training process is shown in the Figure 3. To evaluate the performance of different systems, we use official evaluation methods, *i.e.* we calculate AUC and pAUC to evaluate the performance of each system.

The generator and classifiers were trained on the IDMT-ISA-ELECTRIC-ENGINE dataset and the development dataset, respectively. The train set was firstly spilt into the train (90%) and validate (10%) set. The classifiers were trained on the train set in maximum 100 epochs. The validation set determined the early stopping of the training, i.e., the training would be stopped if its loss failed to decrease in continuous 5 epochs. We used Adam optimizer and set β_1 and β_2 to 0.9 and 0.999, and the initial learning rate was set 10^{-3}.

4.1. Simulate Anomalous Audio Data

Generator was trained on the IDMT-ISA-ELECTRIC-ENGINE dataset, from which we integrated the train data and test data as the training data. And the KR-loss is used as the loss function to train generator. After training, we use the abnormal sound simulation algorithm shown in Figure 1 to simulate anomalous data.

4.2. Autoencoder-based Classifier

We concatenate normal dataset and simulated anomalous dataset as training dataset. Only the development dataset is used as the training dataset. Then the training method is as described in Section 3.2.

4.3. Calculate Anomaly Score

For the AE-based classifier, we calculate the mean square error as the anomaly score. For the CNN-based classifier, we calculate the sigmoid mapping of mean square error as the anomaly score.

5. RESULTS

The performance of the proposed systems above is shown in Tables 5.1 and 5.2. It is found that the dense-based and CNN-based autoencoder classifiers can both lead to better performance than the baseline system, in terms of AUC and pAUC.

Table 5.1: Performance of **"**Wen-UESTC-task2-1" dense-based autoencoder classifier

Type	AUC	pAUC
Toy car	0.753333	0.630127
Toy conveyor	0.770879	0.627321
Fan	0.665770	0.516949
Pump	0.733256	0.608876
Slider	0.914237	0.746218
Valve	0.819897	0.543894

Table 5.2: Performance of **"**Wen-UESTC-task2-2" CNN-based autoencoder classifier

Type	AUC	pAUC
Toy car	0.829205	0.690493
Toy conveyor	0.794139	0.648357
Fan	0.690000	0.531957
Pump	0.752254	0.619995
Slider	0.806795	0.579188
Valve	0.672531	0.509965

Comparing Tables 5.1 and 5.2, we can find that the dense-based autoencoder classifier performs better in types of "slider" and "valve". The CNN-based autoencoder classifier performs better in the other four types. Therefore, it is likely to get higher results with an ensemble of different classifiers. The results of the ensemble model are shown in the Table 5.3. The total average AUC and pAUC of the six types are as high as 0.8 and 0.63, respectively.

Table 5.3: Performance of **"**Wen-UESTC-task2-3" ensemble classifier

Type	AUC	pAUC
Toy car	0.829205	0.690493
Toy conveyor	0.794139	0.648357
Fan	0.690000	0.531957
Pump	0.752254	0.619995
Slider	0.914237	0.746218
Valve	0.819897	0.543894

6. CONCLUSION

This report describes our submissions for DCASE2020 Task 2. With the help of abnormal sound simulation algorithm and autoencoder classifiers, the performance of our proposed system is much improved as compared to the baseline system.

7. REFERENCES

[1] http://dcase.community/workshop2020/.

[2] Y.Koizumi, S.Saito, H. Uematsu, Y.Kawachi and N.Harada, *Unsupervised Detection of Anomalous Sound based on Deep Learning and the Neyman-Pearson Lemma, in Proc. IEEE,* Oct. 2018.

[3] Y. Koizumi, S. Saito, H. Uematsu and N. Harada, "Optimizing acoustic feature extractor for anomalous sound detection based on Neyman-Pearson lemma," 2017 25th European Signal Processing Conference (EUSIPCO), Kos, 2017, pp. 698-702, doi: 10.23919/EUSIPCO.2017.8081297.

[4] Kingma, Diederik P, and Max Welling. "Auto-Encoding Variational Bayes." (2013). Web.

[5] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Noboru Harada, and Keisuke Imoto. *ToyADMOS: a dataset of miniature-machine operating sounds for anomalous sound detection.* In Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), 308–312. November 2019.

[6] Harsh Purohit, Ryo Tanabe, Takeshi Ichige, Takashi Endo, Yuki Nikaido, Kaori Suefusa, and Yohei Kawaguchi. *MIMII Dataset: sound dataset for malfunctioning industrial machine investigation and inspection.* In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019), 209–213. November 2019.

[7] Press, WH; Teukolsky, SA; Vetterling, WT; Flannery, BP (2007). "Section 16.1. Gaussian Mixture Models and k-Means Clustering". Numerical Recipes: The Art of Scientific Computing (3rd ed.). New York: Cambridge University Press. ISBN 978-0-521-88068-8.

[8] Dempster, A.P.; Laird, N.M.; Rubin, D.B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm". Journal of the Royal Statistical Society, Series B. 39 (1): 1–38. JSTOR 2984875. MR 0501537.

[9] Welling, Max; Kingma, Diederik P. (2019). "An Introduction to Variational Autoencoders". Foundations and Trends in Machine Learning. 12 (4): 307–392. arXiv:1906.02691. Bibcode:2019arXiv190602691K. doi:10.1561/2200000056.

[10] Sakurada, M., & Yairi, T. (2014, December). *Anomaly detection using autoencoders with nonlinear dimensionality reduction*. In Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis (p. 4). ACM.

[11] J. An and S. Cho, "Variational Autoencoder based Anomaly Detection using Reconstruction Probability," Technical Report. SNU Data Mining Center, pp.1–18, 2015.

[12] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," Pattern Recognition, vol. 30, no. 7, pp.1145–1159, 1997.