COLOR HALFTONING AND ACOUSTIC ANOMALY DETECTION FOR PRINTING SYSTEMS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Chin-Ning Chen

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

This thesis is dedicated for my family.

ACKNOWLEDGMENTS

Thanks to the most important person in Purdue University, my advisor, Professor Jan. P. Allebach. Without him, I can't keep pursuing the PhD degree. I remembered so clear that the day I joined his team, he said that although my resume shows lots of thing, but the only one he cares is: do I have passion? At that moment, I knew that I finally found a good advisor who will see not only just the final result but also your effort. Thank you so much to pull me out from the darkest nightmare to the bright side. I always remind myself that even you know nothing about your current work, but the most important thing is to be passionate and willing to learn.

Thanks to my committee members, Professor Zoltowski, Professor Maggie Zhu, and Professor Chiu. They give me the different point of view to see the problem and point out the blind side that I never see. Thanks to EISL lab, I learn a lot and always have a good discussion with lab members no matter related to research or to life.

Thanks to my father, Yen-Tsung Chen, my mother Fu-Jung Yeh, and my brother, Zhang-Guan Chen. The endless support and unconditional love is the only thing to motivate myself not to easily give up even facing the worst case. Thanks to my friend, Chia-Jung Chang, you are just like my another family who takes good care of me. I really appreciate all your effort.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

ABBREVIATIONS

FSED	Floyd-Steinberg Error Diffusion
TDFED	Tone-Dependent Fast Error Diffusion
NPs	Neugebauer Primaries
ASR	Automatic Speech Recognition
AED	Acoustic Event Detection
ASC	Acoustic Scene Classification
SED	Sound Event Detection
MFCCs	Mel Frequency Cepstral Coefficients
EMDA	Equalized Mixture Data Augmentation
PSD	Power Spectrum Density
AM	Amplitude Modulation
PCA	Principal Component Analysis
MFC	Mel Frequency Cepstrum
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
OCSVM	One Class Support Vector Machine
SVM	Support Vector Machine
RBF	Radial Basis Function
RF	Random Forest
K-NN	K-Nearest Neighbor
ppm	page per minute
STFT	Short Time Fourier Transform
TFR	Time Frequency Representation
TN	True Negative

TP	True Positive
FN	False Negative
FP	False Positive
P	Precision
R	Recall

ABSTRACT

Chen, Chin-Ning Ph.D., Purdue University, August 2020. Color Halftoning and Acoustic Anomaly Detection for Printing Systems. Major Professor: Jan. P. Allebach.

In the first chapter, we illustrate a big picture of the printing systems and the concentration of this dissertation.

In the second chapter, we present a tone-dependent fast error diffusion algorithm for color images, in which the quantizer is based on a simulated linearized printer space and the filter weight function depends on the ratio of the luminance of the current pixel to the maximum luminance value. The pixels are processed according to a serpentine scan instead of the classic raster scan. We compare the results of our algorithm to those achieved using the fixed Floyd-Steinberg weights and processing the image according to a raster scan ordering.

In the third chapter, we first design a defect generator to generate the synthetic abnormal printer sounds, and then develop or explore three features for sound-based anomaly detection. In the fourth chapter, we explore six classifiers as our anomaly detection models, and explore or develop six augmentation methods to see whether or not an augmented dataset can improve the model performance. In the fifth chapter, we illustrate the data arrangement and the evaluation methods. Finally, we show the evaluation results based on different inputs, different features, and different classifiers.

In the last chapter, we summarize the contributions of this dissertation.

1. INTROCUTION

Printing systems are important in our life. From a small A4 size paper documents to a large size posters along the highway, they all need printing system to be realized. There are lots of printing systems such as office and home printers, commercial printers, and 3D printers which are popular recently. 3D printing is a process for making a physical object from a three-dimensional digital model, typically by laying down many successive thin layers of a material. 3D printing is out of scope in this dissertation, so we will focus on the most common ones: home and office printers.

The difference between home and office printers and commercial printers is that home printers refer to the actual piece of hardware including laser printers and ink jet printers. A digital file is sent to a printer connected to a computer and the printed page is available in a short while. On the other hand, the commercial printers is a business that are printing professionals such as magazines. Usually, it is a print shop with a team of people, and printers for digital printing and also printing presses for offset lithography and other commercial printing processes.

For home printers, as we mentioned, include laser and inkjet printers. The difference between laser and inkjet printers is as follow: Inkjet printing was developed in 1951. It takes a traditional ink cartridge to eject droplets of ink onto paper, and it usually contain black, cyan, magenta, and yellow ink. Inkjet printers can work well for photos and documents that are image-heavy. Generally, they are lighter and smaller than laser printers. On the other hand, the laser printer was invented at Xerox in 1969. It melts toner powder onto paper to create a print, and it is able to handle high print volumes with faster print speeds. The advantage of a laser printer is to produce the perfect copy. For example, the laser electrophotographic process allows the printed black text to look sharp and neat. However, laser printers are more expensive than inkjet printers. If you are printing high-quality color images, then inkjet printers will work well. On the other hand, if you are printing text-based documents or medium-quality color images, laser printers will work well. If you mainly print for a photo, recently, photo inkjet printers are also released. Photo inkjet printers are designed to make pictures stand out while offering the opportunity to work with different print sizes. These printers are usually a bit more expensive than the typical inkjet printers because of the type of ink they use and the fact that they are usually both dye and pigment based.

Multi-function and high printing quality printers need better and longer warranty. How to manage the customer service would be another task for companies. Maintain a good customer service not only keep the confidence from our customer to trust us, but also reduce the cost from companies to replace possible malfunction parts.

To sum up, this dissertation is intended to study the printing algorithm for inkjet printer and the customer service for laser electrophotographic printers based on acoustic-based anomaly detection. The dissertation is structured as follow:

- Chapter 2 focus on a novel aperiodic, dispersed-dot halftoning algorithm.
- Chapter 3 develops a novel way to diagnose the printer health.
- Chapter 4 introduces the anomaly detection models and data augmentation methods.
- Chapter 5 shows the evaluation results.
- Chapter 6 summarizes the contributions of this dissertation.

2. VECTOR MULTILEVEL TONE-DEPENDENT ERROR DIFFUSION IN THE YYCXCZ COLOR SPACE

2.1 Introduction

Halftoning remains a critical aspect of contemporary printing devices. These devices increasingly have multilevel capabilities. We develop a vector error diffusion solution that operates in the YyCxCz linearzed uniform color space [1]. Our results are based on a simulated CMY printer that uses inks from an Indigo 7000 press. We assume a 16-level output for each of the C, M, Y channels; so each output pixel consists of a 4 bit word.

The classic error diffusion proposed by Floyd and Steinberg [2] is an algorithm that generates a binary image by processing the continuous-tone image with neighborhood operations moving through the image in raster scan order, quantizing each pixel in the scan line, and diffusing the error ahead to the neighboring pixels. However, this algorithm also generates worm-like artifacts and some visible structure. There are many papers proposing to solve these problems. In this chapter, we are not going to mention how the artifact problems are solved, but focus how the error diffusion is used in a multilevel color context, rather than a monochrome binary context.

The previous works focusing on mutilevel halftoning and color halftoning are [3–7]. Monga et al. [4] and Yu et al. [7] both use a Neugebauer printer model. In [4], the Neugebauer model was used in the middle step by transforming from CMY to YyCxCz first, next to predict the colorimetric response of the printer and then further to train the error diffusion filter to minimize the error metric. In [7], the authors utilized the Neugebauer model in RGB color space by using trilinear interpolation based on Neugebauer primaries to represent an arbitrary input color. Our method operates with gamut mapping [8, 9] first and then quantizes the input pixel based on sampling the Neugebauer printer model. Both the gamut mapping and the quantization operate in the YyCxCz color space.

Figure 2.1 shows our system pipeline. The input is an sRGB image; and we transform it from sRGB to CIE XYZ [10] to YyCxCz standing for our source gamut. Also, we are given the destination gamut, which is the Indigo gamut; and then we do the gamut mapping to fit the source gamut into the destination gamut. After gamut mapping, we do tone-dependent fast error diffusion (TDFED), which is also in the YyCxCz color space. Finally, we transform the result of TDFED from YyCxCz back to sRGB to display it.

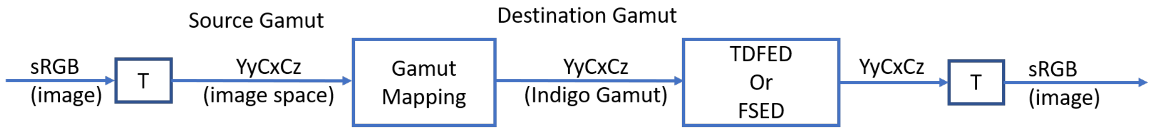


Fig. 2.1.: System pipeline

The rest of parts are organized as follows. Section 2.2 outlines the process of gamut mapping. Section 2.3 first introduce the YyCxCz color space and our printer output space, and then nail into our TDFED algorithm. Also, we will compare our TDFED with classic Floyd-Steinberg Error Diffusion (FSED). Section 2.4 shows and compares the resulting images after gamut mapping, TDFED, and FSED. Finally, Section 2.5 concludes this chapter.

2.2 Gamut Mapping

We want to generate a mapping so that given an sRGB value, we can find its corresponding printable value based on our printer output space. In our system, we are given the destination gamut from the Indigo 7000 press. A brief description of overall process can be split into five part:

Part 1 Soft compress the source lightness from input image to match the destination lightness.

Part 2 Let the source neutral axis and destination neutral axis to align with the Yy axis.

Part 3 Soft compress the source chroma to fit it into the bounding cylinder constructed by the destination gamut. Unfortunately, so far, the source gamut is not actually fit into the destination gamut at every hue angle.

Part 4 We do central compression for lightness and chroma on source gamut to let it actually fit into the destination gamut within each hue sector.

Part 5 Finally, we rotate and shift to move the source gamut to the destination gamut.

The details of the gamut mapping can be seen in Figure 2.2. Also, refer to [8,9] .

2.3 Tone-Dependent Fast Error Diffusion

Our motivation to use tone-dependent fast error diffusion (TDFED) is based on the comparison between Floyd-Steinberg algorithm and TDFED in binary image, as shown in [3]. There are obvious artifacts in the sky in the figure generated by Floyd-Steinberg but the figure generated by TDFED does not have this problem. As a result, we try to use TDFED system on the color image.

2.3.1 YyCxCz Color Space

Before we go to the details of TDFED, here we explain the relation between different color spaces and the reason why we operate in the YyCxCz color space.

The YyCxCz color space is the linearized CIE Lab color space which can be obtained from the CIE XYZ color space, as shown in Equations (2.1), (2.2), (2.3)

$$Yy = 116\frac{Y}{Y_n} - 16 \tag{2.1}$$

$$Cx = 500(\frac{X}{X_n} - \frac{Y}{Y_n}) \tag{2.2}$$

$$Cz = 200(\frac{Y}{Y_n} - \frac{Z}{Z_n}) \tag{2.3}$$

The Yy value represents the luminance, and the Cx and Cz values are the red-green and blue-yellow chrominance components, respectively. Because YyCxCz is a linearized transformation, it can overcome the distortion problem mentioned in [1]

2.3.2 Printer Output Space

We assume a CMY printer with 16 levels per output channel, i.e., 12 bits/pixel. We simulate the printer output space by uniformly interpolating the space defined by the measured 8 Neugebauer Primaries (NPs) which are: W, Y, C, CY, M, MY, CM, CMY. The following steps describe how we simulate our printer output space: First, we choose our target output device, which is the Indigo 7000 press. Second, we print a constant-tone patch for each of the 8 NPs. Third, we measure their CIE XYZ value by using hardware X-rite DTP70 and then convert the obtained XYZ value into YyCxCz value. Next, we uniformly sample $16 \times 16 \times 16$ points along each edge of the CMY color cube as shown in Figure 2.3. Finally, we use tetrahedral interpolation to find their corresponding YyCxCz values as shown in Figure 2.4.

2.3.3 Tone-Dependent Fast Error Diffusion

Figure 2.5 illustrate the block diagram of the TDFED system. In this system, $\vec{f}[m,n]$ is the input image vector, $\tilde{\vec{f}}[m,n]$ is the modified input, and $\vec{g}[m,n]$ is the YyCxCz image vector quantized according to Equation (2.4) where the set the C is the $16 \times 16 \times 16$ set of Neugebauer Primaries for the output device. This means that the current pixel will be replaced by the one of the sampled Neugebauer Primary with the shortest between them. After obtaining $\vec{g}[m,n]$, we can calculate the error vector according to Equation (2.5). Here, the weight function $W(Y_y)$ depends on the position of the pixel to which the error is being diffused relative to that of the pixel being processed and the ratio of the Yy value of the current pixel to the maxmimum Yy value. Plots of the weight function are shown in Figure 2.6, where the star sign represents the current pixel. Moreover, the processing order of TDFED is according to a serpentine scan.

$$\vec{g}[m,n] = \underset{c \in C}{\arg\min}\{\|c - \tilde{\vec{f}}[m,n]\|\} \tag{2.4}$$

$$\vec{e}[m,n] = \vec{g}[m,n] - \tilde{\vec{f}}[m,n] \tag{2.5}$$

2.3.4 Comparison with Floyd-Steinberg Error Diffusion

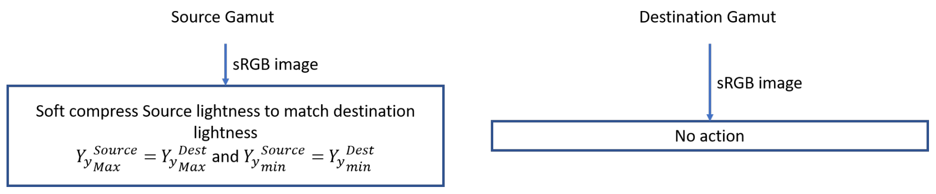
Figure 2.7 illustrates the block diagram of the FSED system. The overall system is similar with TDFED but there are two differences. First, the weight filter H in FSED is the fixed set of numbers where w1 is equal to $\frac{7}{16}$, w2 is equal to $\frac{3}{16}$, w3 is equal to $\frac{5}{16}$, and w4 is equal to $\frac{1}{16}$. In the TDFED system, the weight filter *W(*Y_y*)* is a function. Second, the processing order of FSED is according to raster scan instead of the serpentine scan in the TDFED system. In the experimental result, the comparisons are between FSED with raster scan and TDFED with serpentine scan.

2.4 Experimental Result

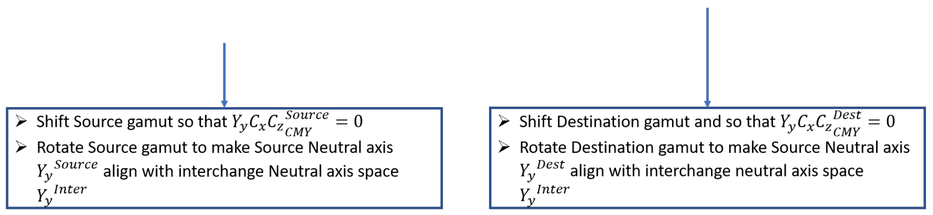
Figure 2.8 shows the comparison of the original image, the gamut mapped image, the TDFED image, and the FSED image. For the gamut mapped image, it is a little bit brighter than the original image. But overall, it looks good. The color rendering results of the TDFED image and the FSED image look very similar at this scale. So we need to zoom in to see the detail of the halftone images. Figure 2.9 shows part of the figure in larger scale to have the big picture of what is generated by halftoning. Figures 2.10 to 2.12 shows the image detail in pixel scale. In FSED image, there are obvious checkerboard and texture cliques artifact. But in some case as shown in Figure 2.12, FSED performs better than TDFED. However, the overall image quality generated by TDFED is better than that of FSED.

2.5 Conclusion

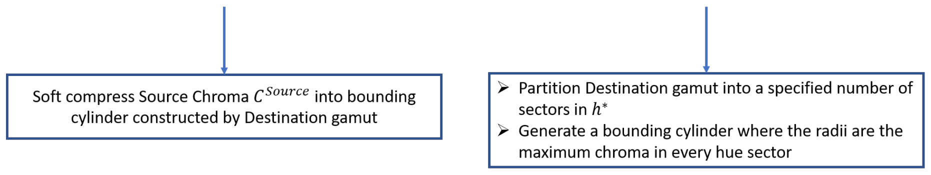
We have developed a halftoning solution for a simulated multilevel CMY printer with 16 levels per channel. Our proposed imaging pipeline consists of: First, transform input sRGB to YyCxCz linearized uniform color space. Next, we do the gamut mapping to fit the source gamut into the printer gamut also in the YyCxCz color space. Finally, we do vector error diffusion that maps modified continuous-tone image values to the nearest output color on a $16 \times 16 \times 16$ grid. It yields good quality output images but the improvement over FSED provided by TDFED is not as significant as it is for a binary output device.


Source Gamut
Destination Gamut
sRGB image
sRGB image
Soft compress Source lightness to match destination lightness $Y_{y_{Max}}^{Source} = Y_{y_{Max}}^{Dest}$ and $Y_{y_{min}}^{Source} = Y_{y_{min}}^{Dest}$
No action

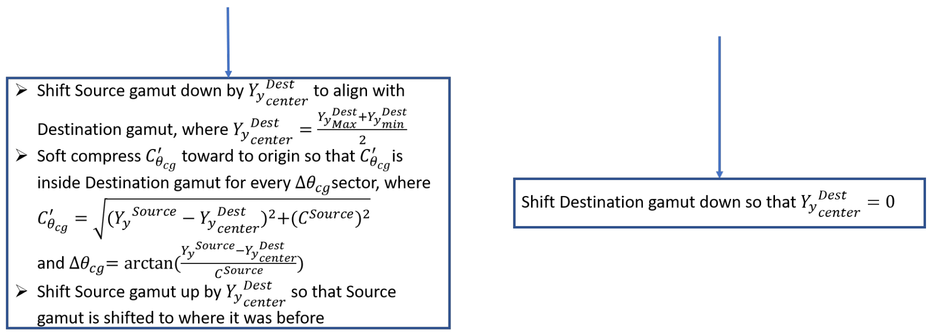
(a) Part 1


Shift Source gamut so that $Y_yC_xC_{z_{CMY}}^{Source} = 0$
Rotate Source gamut to make Source Neutral axis Y_y^{Source} align with interchange Neutral axis space Y_y^{Inter}
Shift Destination gamut and so that $Y_yC_xC_{z_{CMY}}^{Dest} = 0$
Rotate Destination gamut to make Source Neutral axis Y_y^{Dest} align with interchange neutral axis space Y_y^{Inter}

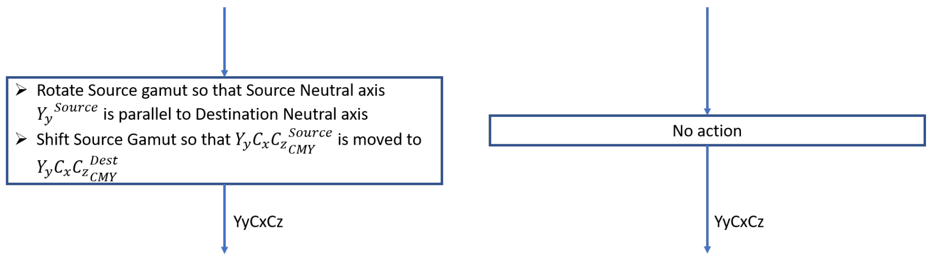
(b) Part 2


Soft compress Source Chroma C^{Source} into bounding cylinder constructed by Destination gamut
Partition Destination gamut into a specified number of sectors in h^*
Generate a bounding cylinder where the radii are the maximum chroma in every hue sector

(c) Part 3


Shift Source gamut down by $Y_{y_{center}}^{Dest}$ to align with Destination gamut, where $Y_{y_{center}}^{Dest} = \frac{Y_{y_{Max}}^{Dest} + Y_{y_{min}}^{Dest}}{2}$
Soft compress $C'_{\theta_{cg}}$ toward to origin so that $C'_{\theta_{cg}}$ is inside Destination gamut for every $\Delta\theta_{cg}$ sector, where $C'_{\theta_{cg}} = \sqrt{(Y_y^{Source} - Y_{y_{center}}^{Dest})^2 + (C^{Source})^2}$ and $\Delta\theta_{cg} = \arctan(\frac{Y_y^{Source} - Y_{y_{center}}^{Dest}}{C^{Source}})$
Shift Source gamut up by $Y_{y_{center}}^{Dest}$ so that Source gamut is shifted to where it was before
Shift Destination gamut down so that $Y_{y_{center}}^{Dest} = 0$

(d) Part 4


Rotate Source gamut so that Source Neutral axis Y_y^{Source} is parallel to Destination Neutral axis
Shift Source Gamut so that $Y_yC_xC_{z_{CMY}}^{Source}$ is moved to $Y_yC_xC_{z_{CMY}}^{Dest}$
No action
YyCxCz
YyCxCz

(e) Part 5

Fig. 2.2.: Process of gamut mapping. Left hand side is the action for source gamut and the right hand side action is for destination gamut. (a) Part 1 (b) Part 2 (c) Part 3 (d) Part 4 (e) Part 5

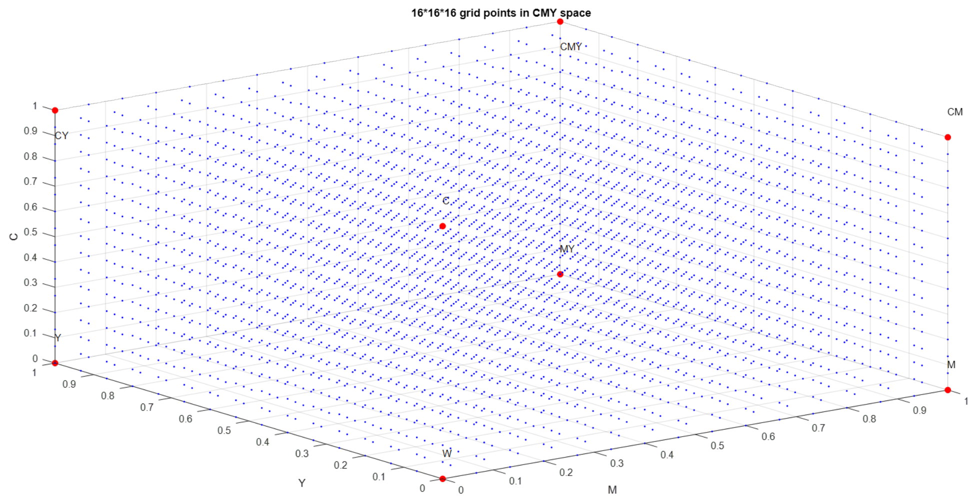


Fig. 2.3.: Sampled points in CMY color space

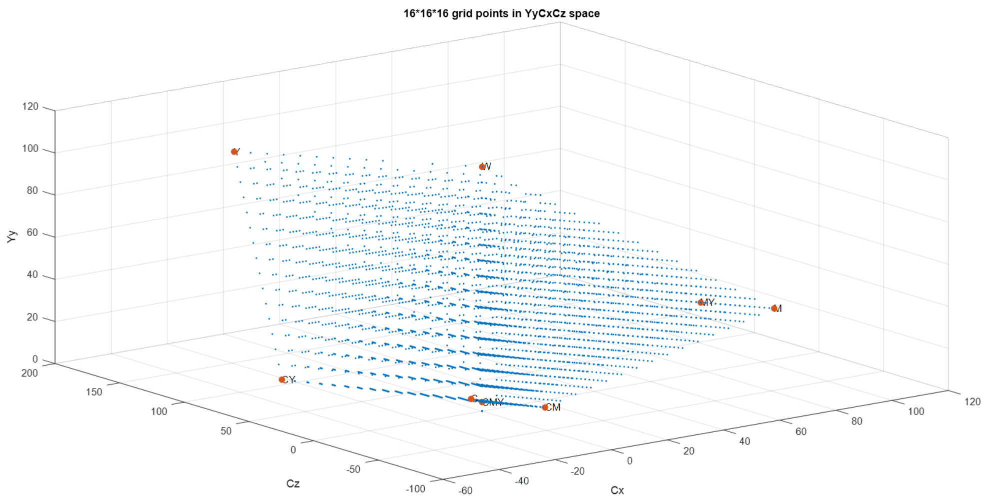


Fig. 2.4.: Sampled points in YyCxCz color space

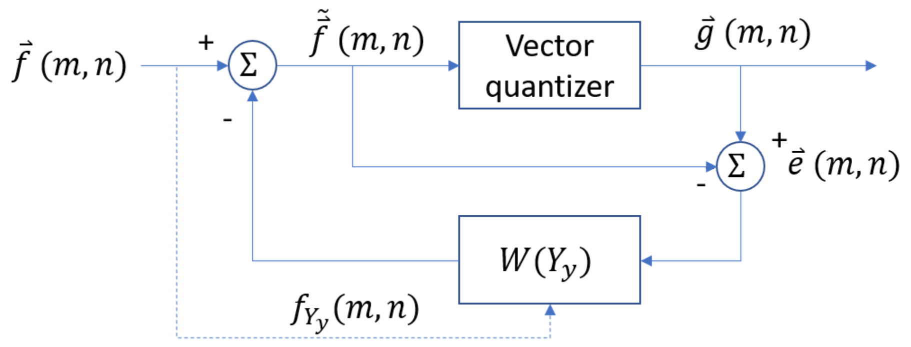


Fig. 2.5.: Tone-dependent fast error diffusion system

	*	w1
w2	w3	w4

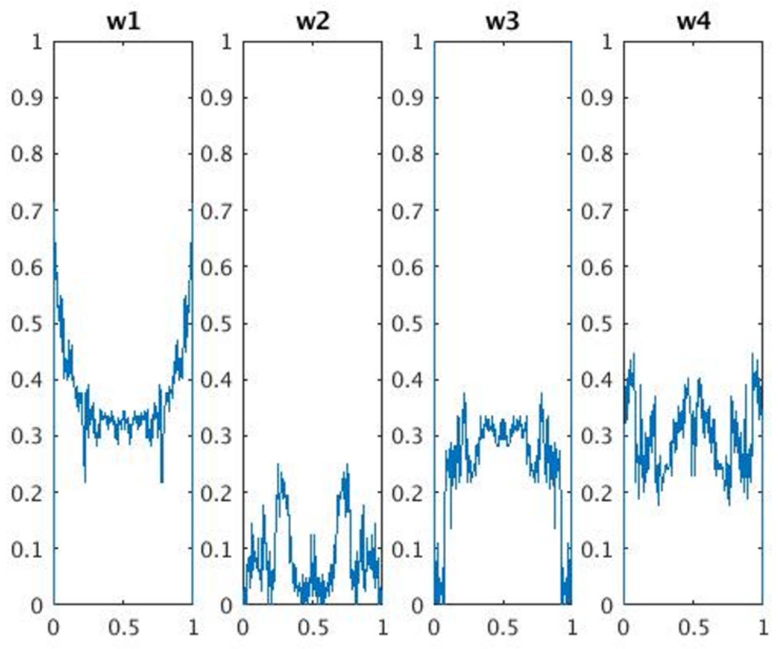


Fig. 2.6.: weight function for TDFED

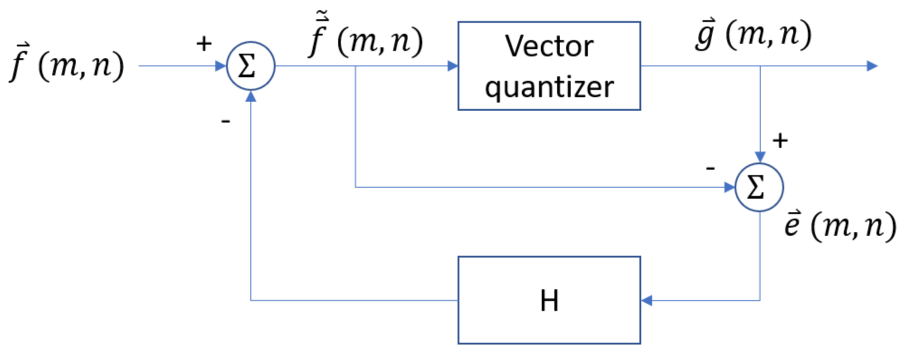


Fig. 2.7.: Floyd-Steinber error diffusion system

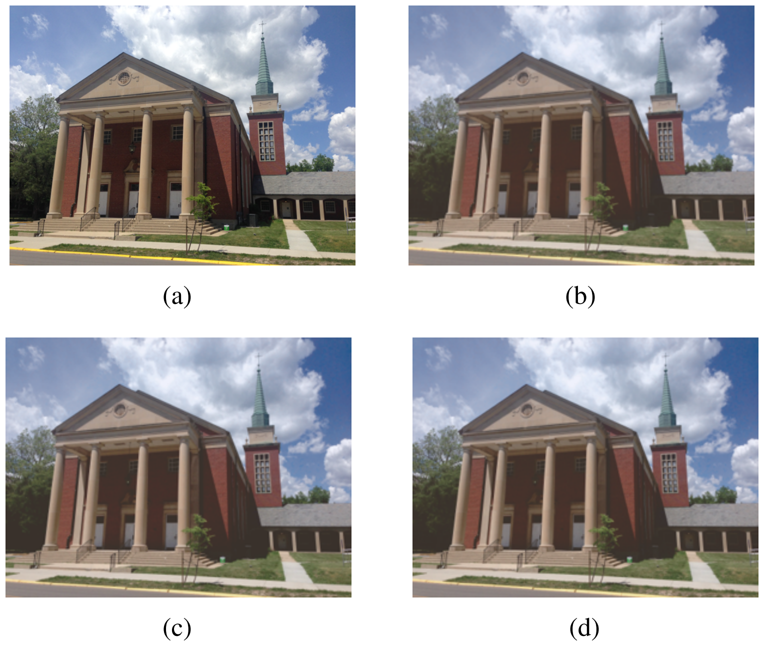
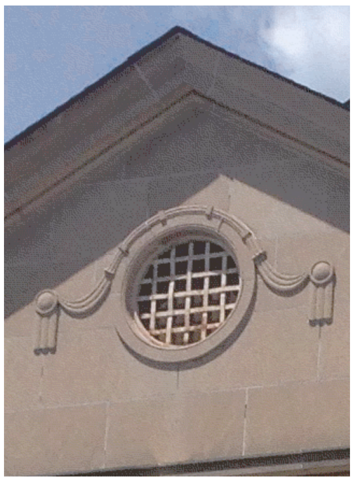


Fig. 2.8.: Comparison of resulting image at three stages in the system pipeline. (a) Original image (b) Gamut mapped image (c) TDFED image (d) FSED image


(a)

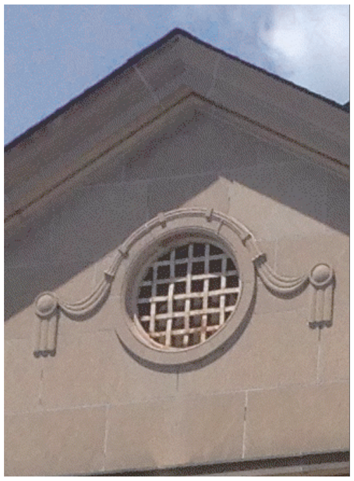
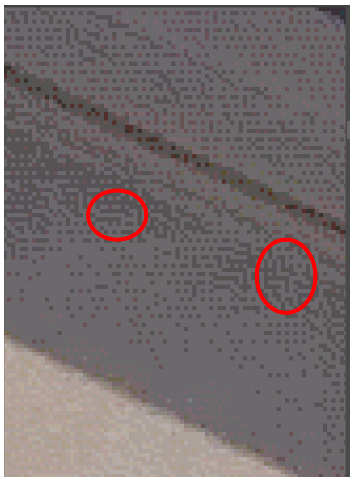

(b)

Fig. 2.9.: The result zoom in for Figure 2.8 (c) and (d) respectively: (a) TDFED image (b) FSED image.


(a)

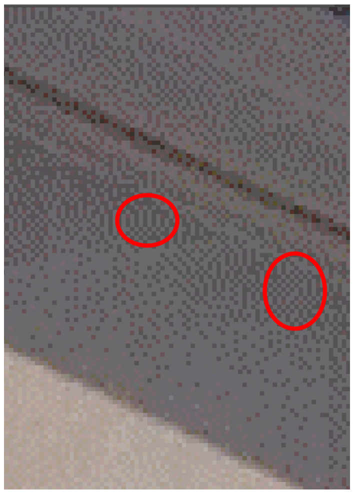

(b)

Fig. 2.10.: Comparison between (a) TDFED image and (b) FSED image. We can see the obvious checkerboard artifact in FSED.

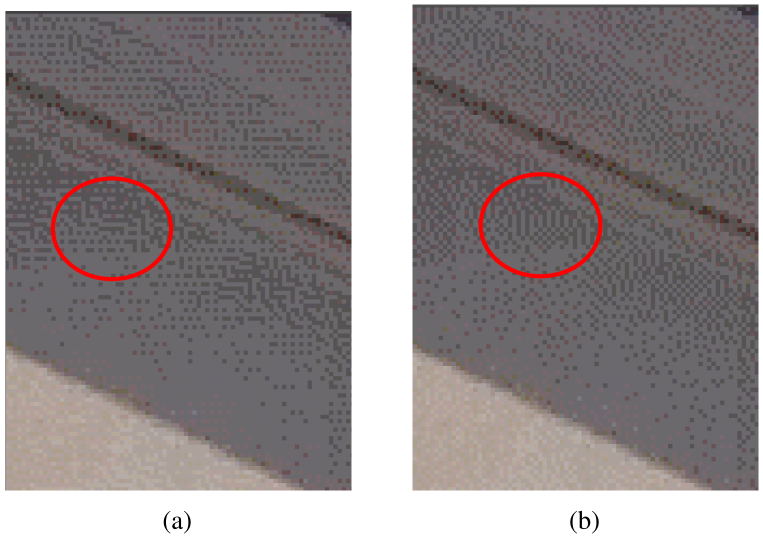


Fig. 2.11.: Comparison between (a) TDFED image and (b) FSED image. We can see the texture cliques artifact in FSED.

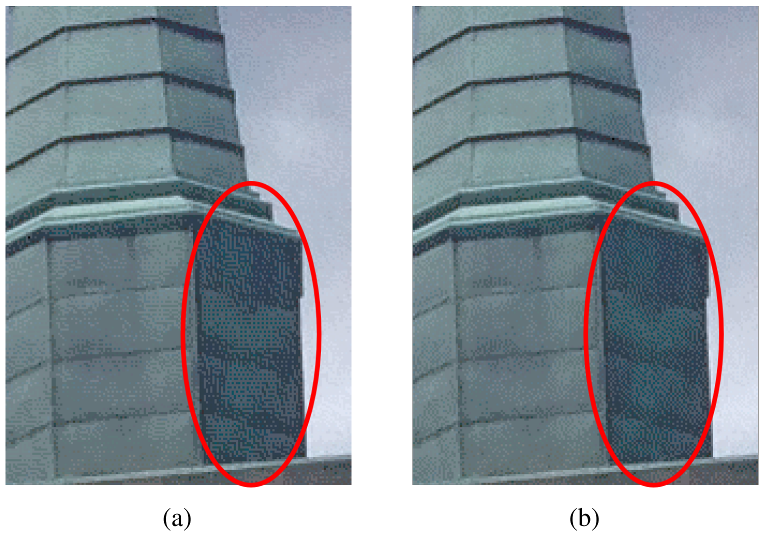


Fig. 2.12.: Comparison between (a) TDFED image and (b) FSED image. We can see that in this case, FSED performs better than TDFED.

3. FEATURES FOR SOUND-BASED ANOMALY DETECTION

3.1 Introduction

Anomaly detection is used in a variety of applications such as fraud detection for credit card [11], security systems [12], and machine operational condition. Anomaly has different definitions in different cases, but generally speaking, a behavior, which has obviously different pattern compared with the normal behavior that was well defined by the system, is called anomaly. In this chapter, we focus on the sound-based anomaly detection. Sound-based system have been attracting more attention because of inexpensive microphone settings and recordings [13]. Recent researches have also explored plenty of techniques for modeling the acoustic signal in order to better capture its important features.

Basically, there are three types of sound-based system: supervised, semi-supervised, and unsupervised. The key difference between them is the use of the labeled training data. Supervised system uses labeled data for both normal and anomalous classes during training process. Semi-supervised system only uses labeled data for normal class during training process. Finally, unsupervised system doesn't need training data. Supervised system applications include Automatic speech recognition (ASR) [14, 15], acoustic event detection (AED) [16], acoustic scene classification (ASC) [17, 18], and sound event detection (SED) [13]. Semi-supervised and unsupervised applications are mainly focusing on anomaly/outlier detection [19, 20].

The acoustic features are critical to different applications as mentioned previously. Various hand-crafted descriptors have been proposed such as Mel frequency cepstral coefficients (MFCCs) [21], filter bank [22, 23], spectrogram [24, 25], and bag-of-audio-words [26, 27]; and they were modeled with Support Vector Machine (SVM) [28]. But even though we have multiple feature representations to fit the specific application, there still has one problem: the lack of the acoustic data. Unlike image dataset, there are few public dataset that are suitable for various acoustic applications. As a result, based on the limited data, data augmentation plays a critical role to expand the dataset size.

Figure 3.1 illustrates the system pipeline for printer sound-based anomaly detection. Our goal is to use the acoustic signal to diagnose the machine health. The input of the

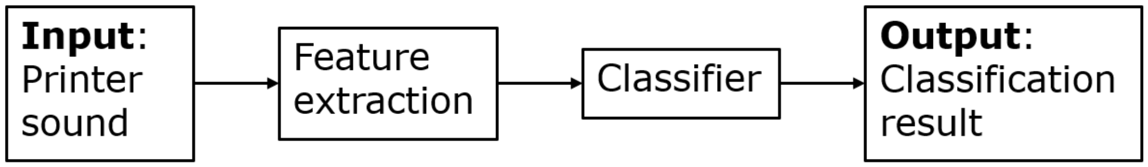


Fig. 3.1.: System pipeline for printer sound-based anomaly detection.

system is printer sounds and the final output of the system is the classification result. The purpose of data augmentation is to increase the number of the data and decrease feature development cycle. We can have possible solution ready at product deployment and confirm with production data, instead of using production data for development. Note that we are using different augmented datasets to train the models and test them with the same real testing data. All of the dataset will go through the same pipeline and we can see the comparison evaluation in Section 5.3.

We briefly introduce the hardware setting and how to collect the printer sound. The detail information can refer [29]. Figure 3.2 shows the printer that we used to collect the sound and its interior structure for the component and the path for the paper that rolling inside the printer. Fig. 3.3 shows the setting of the microphone. Practically, as shown in Figure 3.4, the printer has built-in microphone and detector. When the printer is working, it will automatically generate a extracted feature, which is called feature matrix. The feature matrix will be uploaded to HP cloud and do further process to diagnose whether the printer is in good condition or it has some problem. The reason why the extracted feature matrix is uploaded to HP cloud instead of the sound file is for the security issue. The microphone inside the printer may record something important that you don't want to share.

The remainder of the chapter is organized as follows. Section 3.2 illustrate the first stage feature extraction with the detector and how to synthesize the abnormal acoustic

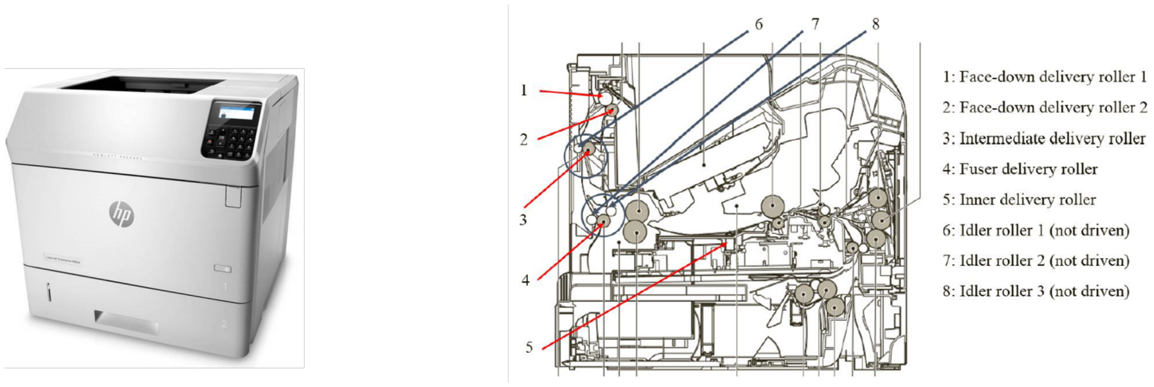

1
2
3
4
5
6
7
8
1: Face-down delivery roller 1
2: Face-down delivery roller 2
3: Intermediate delivery roller
4: Fuser delivery roller
5: Inner delivery roller
6: Idler roller 1 (not driven)
7: Idler roller 2 (not driven)
8: Idler roller 3 (not driven)

Fig. 3.2.: HP Laserjet Managed MFP E82560dn and its interior structure. [29]

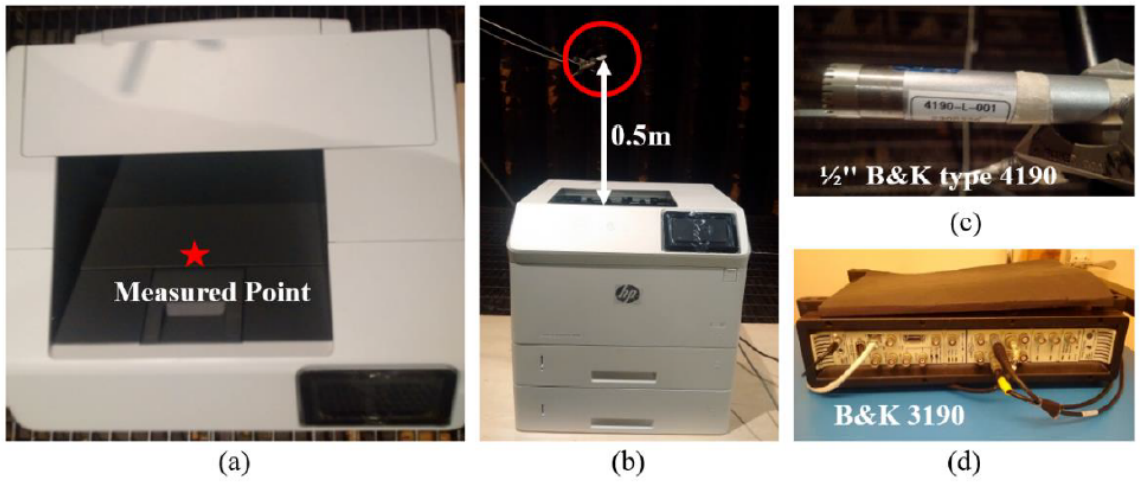


Fig. 3.3.: (a) Measured point top view, (b) location of the microphone, (c) microphone used for recording the printer sound, (d)analog to digital conversion [29]

signals. Section 3.3 illustrates three feature extraction methods. Section 4.1 reviews four classifiers. Section 4.2 illustrates six augmentation methods. Section 5.1 demonstrates the arrangement for training and testing data. Section 5.2 demonstrates the evaluation methods. Finally, Section 5.3 shows the experiment results, and Section 5.4 is conclusion.

3.2 Feature Analysis

We have already known how the anomaly detection system works and the source of the input data, but we still have a problem. The problem is that we only have collected normal

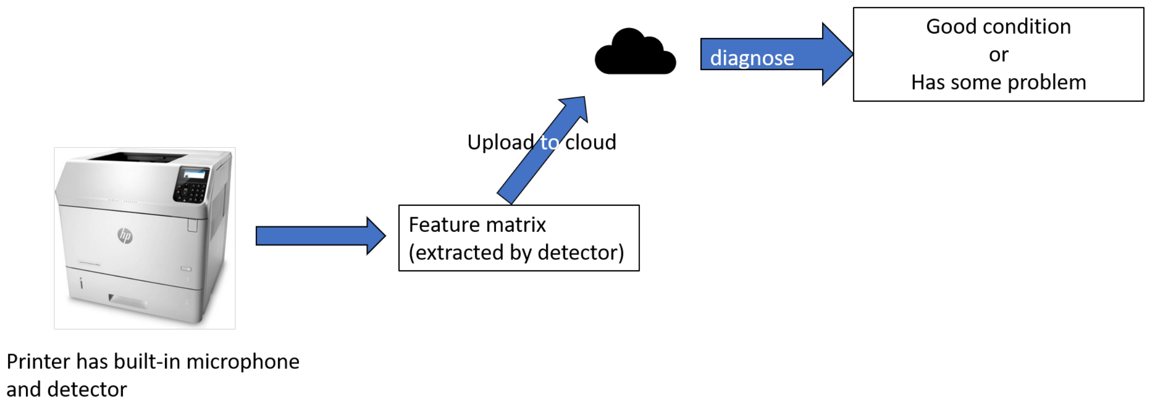
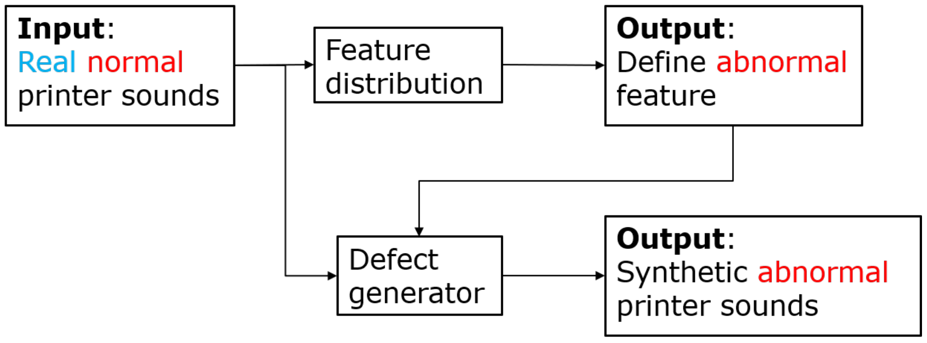


Fig. 3.4.: How to diagnose the printer health.

printer sounds. As a result, the question here would be how to synthesize the abnormal printer sounds.

In this section, first of all, we will introduce the detector [30], which is used for the first stage feature extraction. Second, based on the feature distribution of the normal real printer sounds, we can define what kind of features represent abnormal characteristic. Third, based on feature distribution that we just defined in the second part, we can artificially synthesize the abnormal printer sound. Figure 3.5 illustrates the pipeline for how to synthesize the abnormal printer sounds.



Input: Real normal printer sounds
Feature distribution
Output: Define abnormal feature
Defect generator
Output: Synthetic abnormal printer sounds

Fig. 3.5.: System pipeline for data pre-processing

3.2.1 Detector

We use the detector based on [30] as our first stage feature extraction. Figure 3.6 shows the pipeline of the detector. Basically, it is constructed in two parts by strong tone information and modulation information.

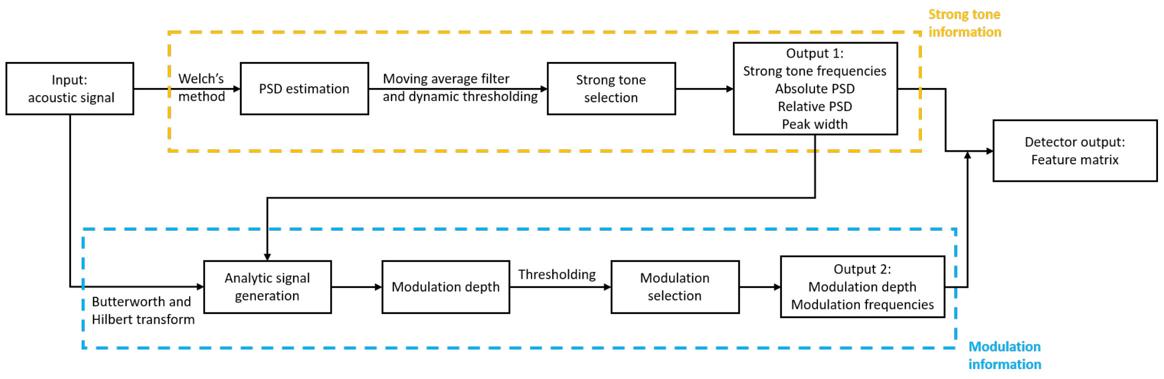

Strong tone information
Input: acoustic signal
Welch's method
PSD estimation
Moving average filter and dynamic thresholding
Strong tone selection
Output 1: Strong tone frequencies Absolute PSD Relative PSD Peak width
Detector output: Feature matrix
Butterworth and Hilbert transform
Analytic signal generation
Modulation depth
Thresholding
Modulation selection
Output 2: Modulation depth Modulation frequencies
Modulation information

Fig. 3.6.: Detector pipeline.

For the strong tone information, we first calculate the power spectrum density (PSD) estimation of the input printer sound by Welch's method [31]. Next, we use a moving average filter to smooth the PSD estimation in the first step and then to find the dynamic threshold as shown in Figure 3.7. The blue curve represents PSD estimation, the green curve represents the smoothed PSD, and the magenta curve is the dynamic threshold based on the green curve. Finally, based on the dynamic threshold (magenta curve), we extract strong tone frequencies, relative PSD, absolute PSD, and peak width. The extracted strong tone frequencies are the blue peaks with peak width larger than 2 as shown in Figure 3.7. The absolute PSD is the value of PSD at the strong tone frequencies. The relative PSD is the difference between absolute PSD (blue curve) and smoothed PSD (green curve) at the strong tone frequencies. Figure 3.8 is an example of the zoomed in version of Figure 3.7 at certain strong tone frequency 11.89 kHz. It explains more about peak width and relative PSD.

With this information for each strong tone, we can step to the modulation information part to generate the analytic signal by using a Butterworth filter and the Hilbert transform

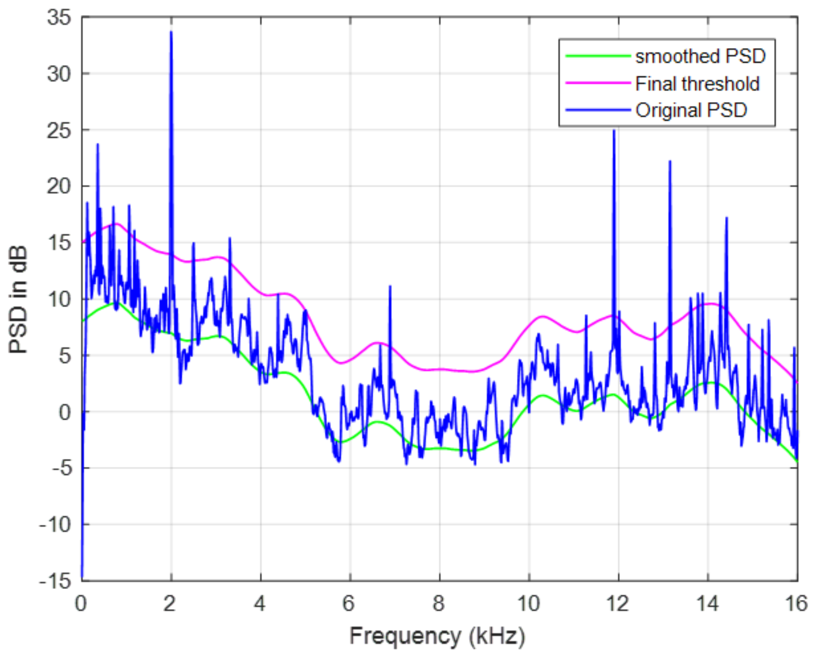


Fig. 3.7.: Power spectrum density (PSD) estimation.

as shown in Figure 3.9. The purpose of a Butterworth filter is to extract the neighbor information which centers at a strong tone frequency, and the purpose of the Hilbert transform is to phase shift 90 degree of the output from a Butterworth filter. We take the output of a Butterworth filter as the real part of the analytic signal and take the output of the Hilbert transform as the imaginary part of the analytic signal. As shown in Figure 3.9, the blue curve is the analytic signal at a strong tone frequency 11.89 kHz and the orange curve is its instantaneous amplitude, which is the absolute value of the analytic signal. Next, based on the generated analytic signal, we can calculate its modulation depth as shown in Equation 3.1,

$$\text{modulation depth} = \frac{\text{PSD of the instantaneous amplitude} \times \text{freqR1}}{\text{PSD of its corresponding strong tone frequency} \times \text{freqR2}} \times 100\% \tag{3.1}$$

where freqR1 is the frequency resolution when we calculate the PSD of the instantaneous amplitude, and freqR2 is the frequency resolution when we calculate the PSD of the input audio signal. If the calculated modulation depth exceeds the threshold line (red line) as

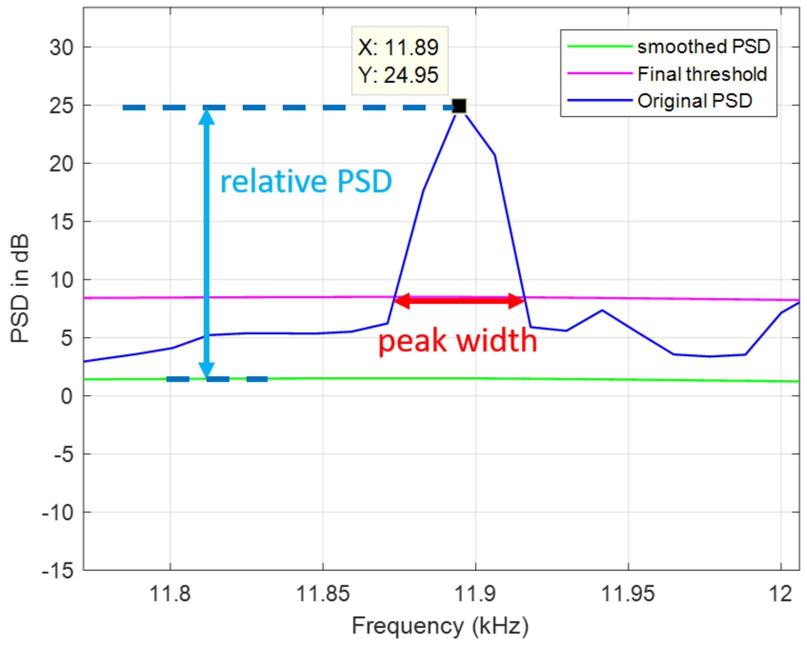


Fig. 3.8.: The information of peak width and relative PSD at strong tone frequency 11.89 kHz.

shown in Figure 3.10, its corresponding modulation frequency and itself will be recorded. Note that the threshold for modulation depth is 10%.

The final output of the detector, which includes strong tone and modulation information, is called the feature matrix as shown in Figure 3.11. The first four columns are the strong tone information: Strong tone frequencies, relative PSD, absolute PSD, and peak width. The last two columns are the modulation information: Modulation frequencies and modulation depth.

3.2.2 Feature Distribution

Because we only have collected real normal printer sounds, we would like to find some characteristic of abnormal features based on the analysis of normal data. Figure 3.12 shows the histogram of the strong tone frequencies from the collected normal printer sounds. Based on the characteristic as shown in Figure 3.12, we will synthesize the synthetic ab-

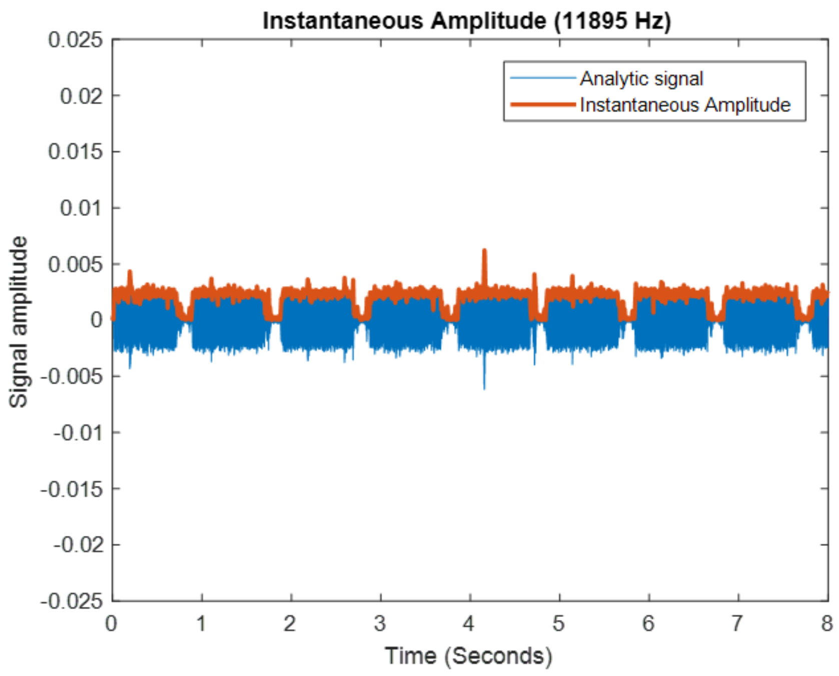


Fig. 3.9.: Example of an analytic signal at strong tone frequency 11.89 kHz.

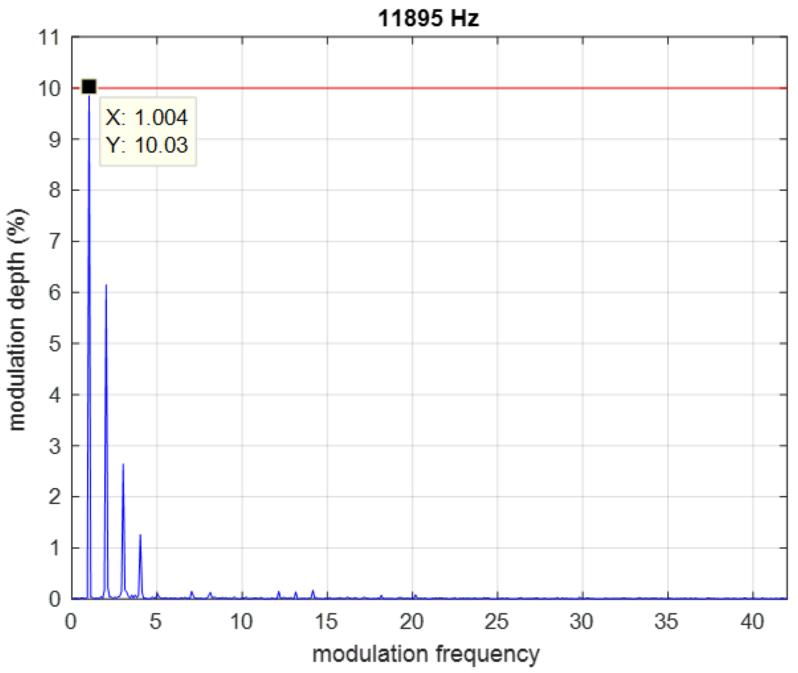
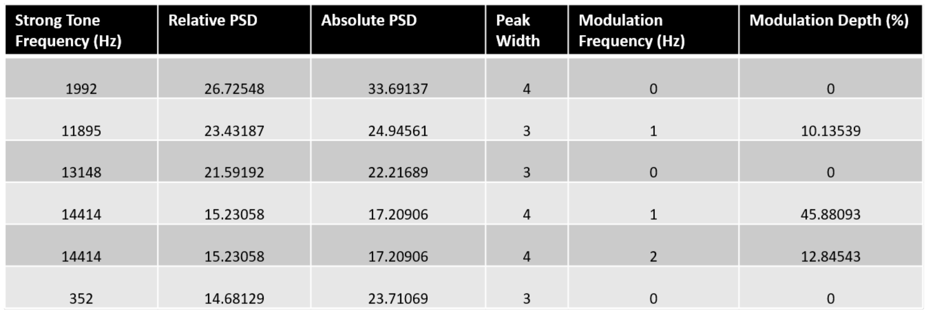


Fig. 3.10.: Modulation information selection at strong tone frequency 11.89 kHz.



Strong Tone Frequency (Hz)	Relative PSD	Absolute PSD	Peak Width	Modulation Frequency (Hz)	Modulation Depth (%)
1992	26.72548	33.69137	4	0	0
11895	23.43187	24.94561	3	1	10.13539
13148	21.59192	22.21689	3	0	0
14414	15.23058	17.20906	4	1	45.88093
14414	15.23058	17.20906	4	2	12.84543
352	14.68129	23.71069	3	0	0

Fig. 3.11.: Example of the feature matrix.

normal printer sounds. We first use two normal distributions to fit the strong tone frequency histogram as shown in Figure 3.13 and then define the frequency ranges that represent abnormal features. One of the property of normal distribution is that $\pm 2\sigma$ contains up to 95.4% of the data. Based on the observation of Figure 3.11 and 3.13, the definition of the abnormal features are: Strong tone frequency ranges from 3 kHz to 10 kHz and modulation depth larger than 100%.

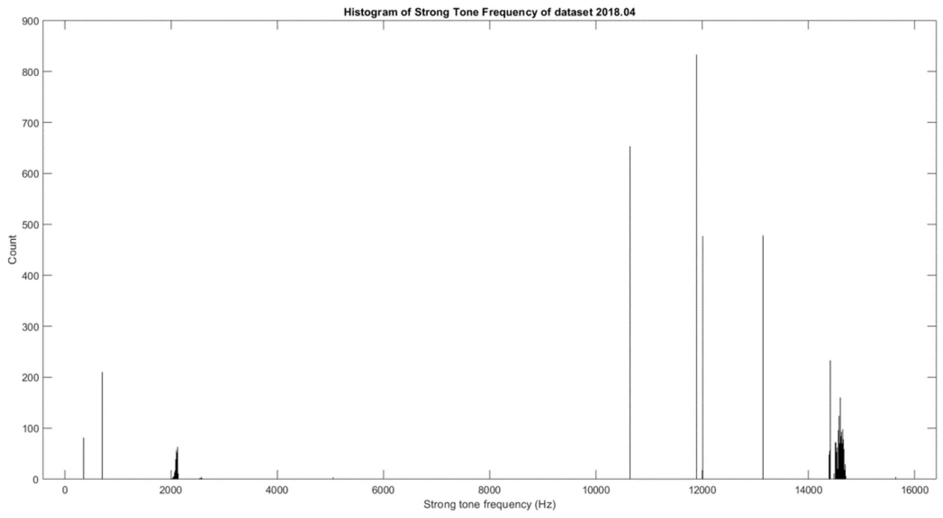
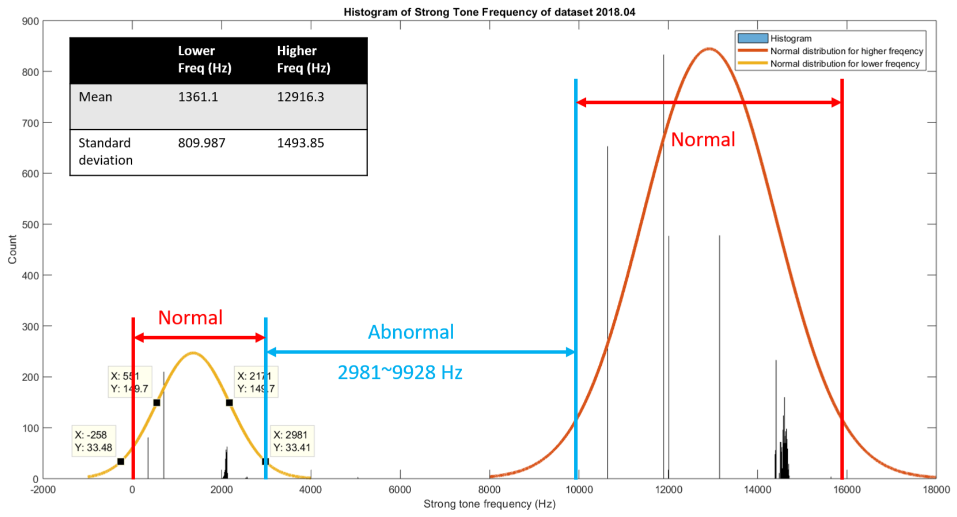


Fig. 3.12.: Histogram of strong tone frequencies from real normal printer sound dataset.



	Lower Freq (Hz)	Higher Freq (Hz)
Mean	1361.1	12916.3
Standard deviation	809.987	1493.85

Fig. 3.13.: Fitting histogram of strong tone frequencies with two normal distributions.

3.2.3 Defect Generator

The purpose of the defect generator is to generate the synthetic abnormal acoustic signal since we don't have real abnormal printer sounds. Inspired by Figure 3.9, we would like to mimic the waveform of the modulating signal. Based on the abnormal features that we defined, the inspiration from Figure 3.9, and the concept of amplitude modulation (AM), we can specify certain abnormal strong tone frequencies as the carrier signal to carry the modulation frequency as the modulating signal as shown in Figure 3.14. Before go through the defect generator, we first review the concept of amplitude modulation [32] and Fourier series representation.

For amplitude modulation, we have two signals: Carrier signal and modulating signal as shown in Equations 3.2 and 3.3

$$c(t) = A_c \cdot sin(2\pi f_c t) \tag{3.2}$$

$$m(t) = A_m \cdot sin(2\pi f_m t) \tag{3.3}$$

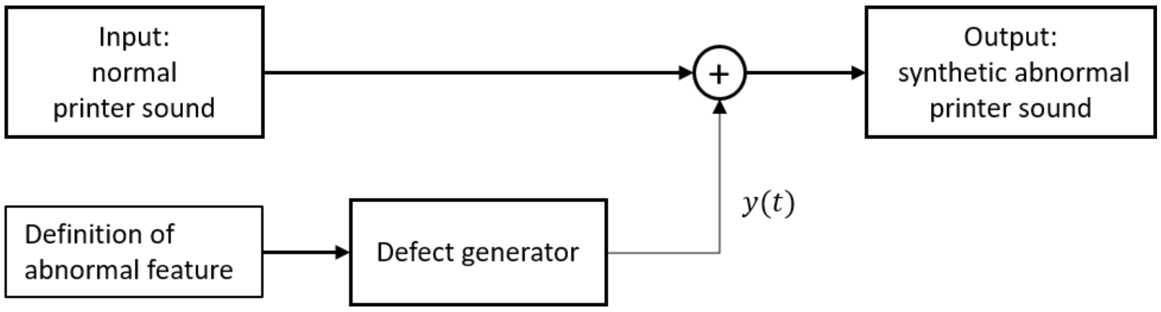


Fig. 3.14.: Defect generator pipeline.

where f_c and f_m are the carrier frequency and the modulating frequency, respectively. The artificial defect can further be written as in Equation 3.4

$$y(t) = \left(1 + \frac{m(t)}{A_c}\right) c(t) = (1 + \mu \cdot sin(2\pi f_m t))\, c(t) \tag{3.4}$$

where μ is the ratio of the amplitude of the modulating signal to the amplitude of carrier signal. In our case, the carrier frequency f_c is the strong tone frequency, and the modulating signal is the modulation frequency generated by the sinusoidal model. The sinusoidal model combines a square wave and one sine wave with specific modulation frequency as shown in Equation 3.3. A square wave can be simulated by multiple sine waves based on Fourier Series representation as shown in Equation 3.5

$$s(t) = a_0 + \sum_{k=1}^{\infty} A_k sin(2\pi k f_0 t) \tag{3.5}$$

where

$$A_k = \begin{cases} \frac{2}{\pi k}, & \text{k is odd.} \\ 0, & \text{k is even.} \end{cases} \tag{3.6}$$

and

$$a_0 = \frac{1}{2},\ f_0 = 1, \text{ and } k = 1, ..., 19 \tag{3.7}$$

Note that the modulating signal is the combination of the square wave and one additional modulation frequency.

We generate the synthetic abnormal printer sounds based on the abnormal features that we defined in Section 3.2.2 and the the concept of amplitude modulation (AM). First of all, we specify the strong tone frequency at 5 kHz as the carrier signal as shown in Figure 3.15 and Equation 3.8

$$c(t) = A_c \cdot sin(2\pi 5000t) \tag{3.8}$$

where A_c is equal to 0.005. Next, one of the carried modulating signal is the simulated

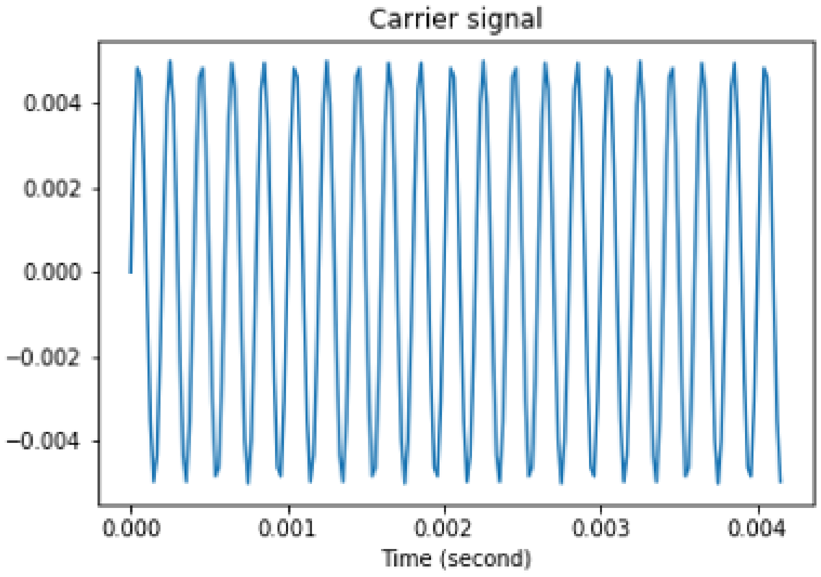


Fig. 3.15.: Carrier signal $c(t)$.

square wave as shown in Figure 3.16 and Equation 3.9

$$m_1(t) = \frac{1}{2} + \sum_{k=1}^{19} \frac{2}{\pi k} sin(2\pi kt) \tag{3.9}$$

The other modulating signal is the carried modulation frequency at 5 Hz as shown in Figure 3.16 and Equation 3.10

$$m_2(t) = \frac{2}{\pi} \cdot sin(2\pi 5t) \tag{3.10}$$

The final modulating signal combines the simulated square wave and the carried modula-

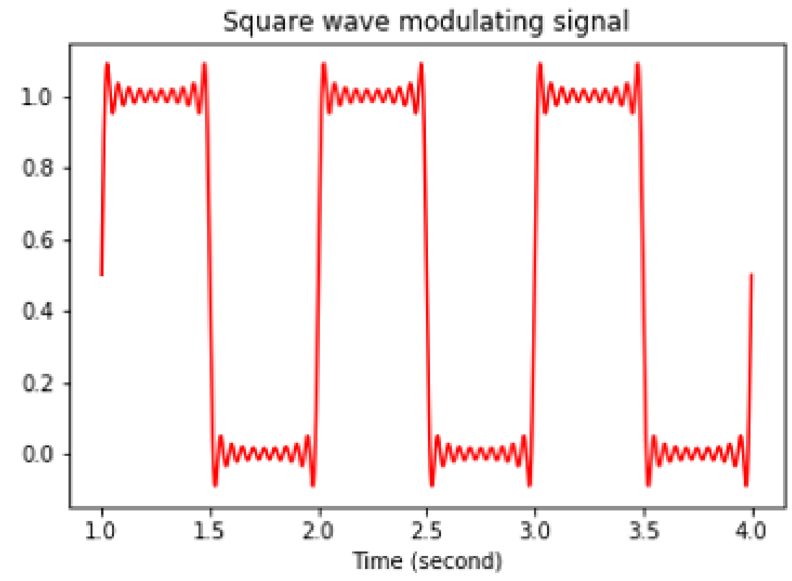


Fig. 3.16.: Simulated square wave $m_1(t)$.

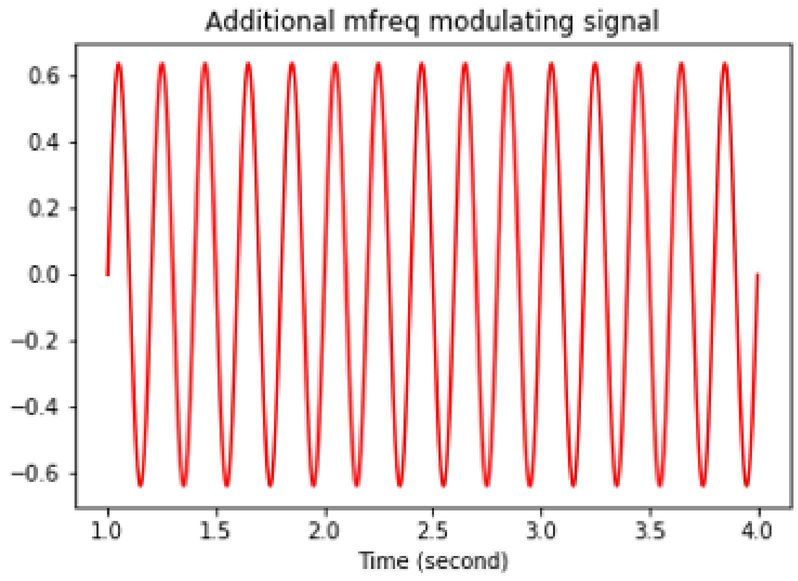


Fig. 3.17.: Carried modulation frequency modulating signal $m_2(t)$.

tion frequency as shown in Figure 3.18 and Equation 3.11

$$m(t) = m_1(t) + m_2(t) \tag{3.11}$$

Finally, the final defect is shown in Figure 3.19 and Equation 3.12

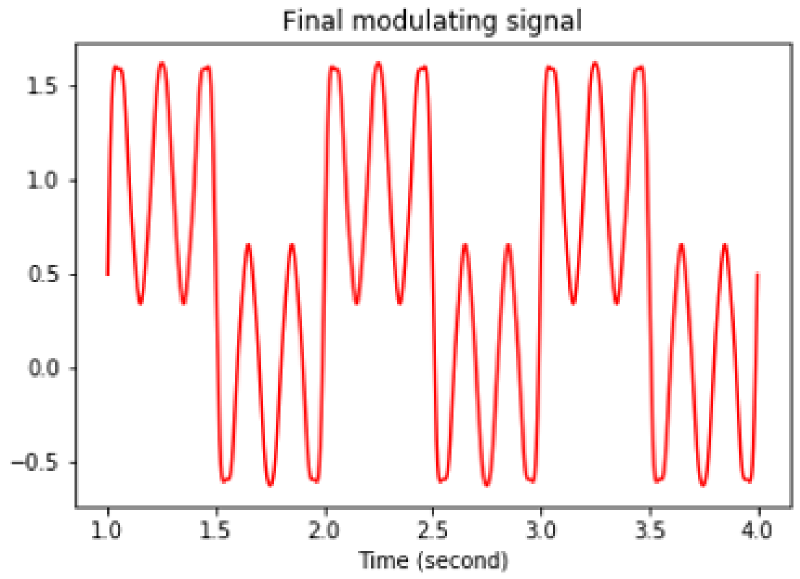


Fig. 3.18.: Final modulating signal $m(t)$.

$$y(t) = (1 + m(t))c(t) \tag{3.12}$$

The blue part is the carrier signal at 5 kHz and the red part is the modulating signal with the combination of the simulated square wave and the carried modulation frequency. To see whether this defect generator works or not, let’s see the feature matrix of the normal acoustic signal and its synthetic abnormal acoustic signal. Figure 3.11 is the feature matrix of the normal acoustic signal, we can see that the strong tone frequencies and modulation depths are land in the range that we define as normal. Compare with Figure 3.20, the feature matrix of the synthetic abnormal acoustic signal contains abnormal features, which is the strong tone frequency at 5 kHz. Figure 3.21 shows the analytic signal of the synthetic abnormal acoustic signal at strong tone frequency 5 kHz. Based on the comparison of the feature matrix, we can justify that our defect generator can successfully synthesize the abnormal acoustic signal.

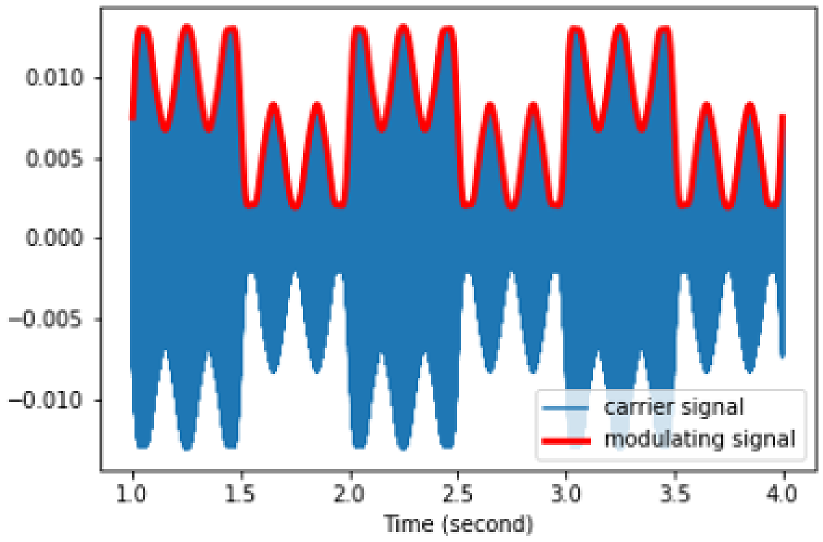
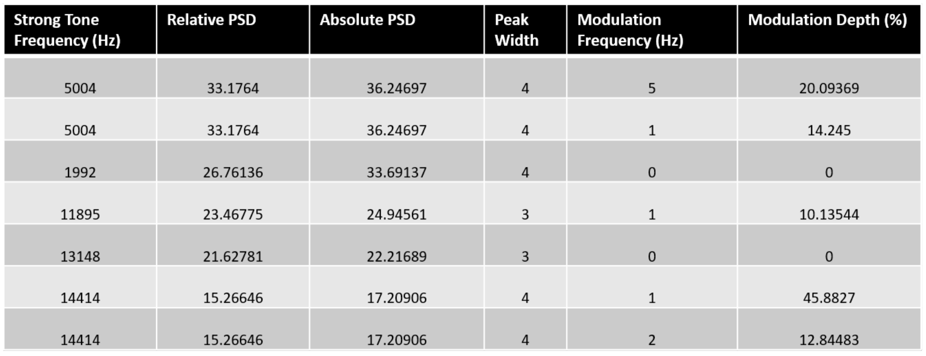


Fig. 3.19.: Final defect $y(t)$.



Strong Tone Frequency (Hz)	Relative PSD	Absolute PSD	Peak Width	Modulation Frequency (Hz)	Modulation Depth (%)
5004	33.1764	36.24697	4	5	20.09369
5004	33.1764	36.24697	4	1	14.245
1992	26.76136	33.69137	4	0	0
11895	23.46775	24.94561	3	1	10.13544
13148	21.62781	22.21689	3	0	0
14414	15.26646	17.20906	4	1	45.8827
14414	15.26646	17.20906	4	2	12.84483

Fig. 3.20.: Feature matrix of synthetic abnormal acoustic signal.

3.3 Feature Extraction

In this section, we will introduce three features: Detector with principal component analysis, Mel frequency cepstral coefficients (MFCCs), and detector with mean and standard deviation.

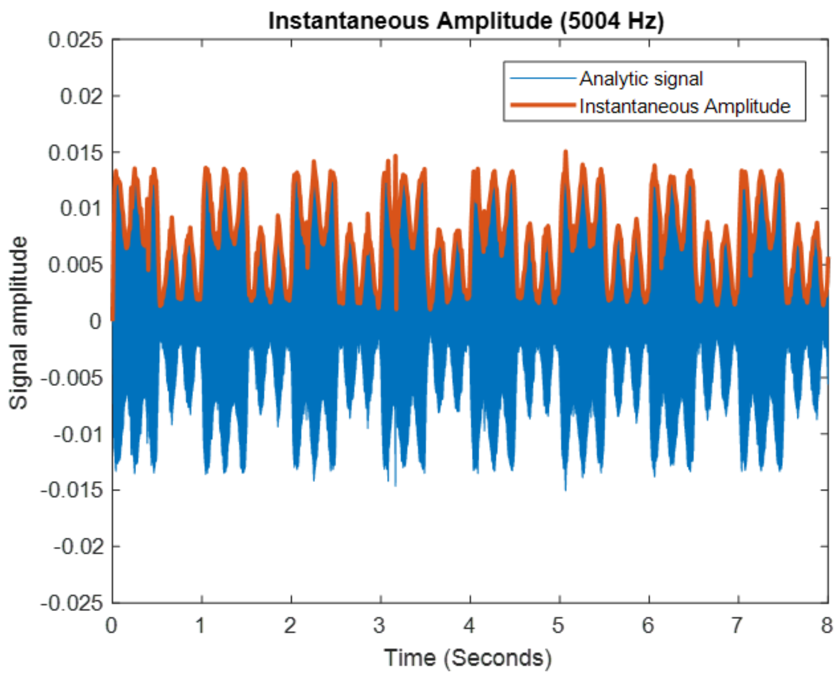


Fig. 3.21.: Analytic signal at strong tone frequency 5 kHz

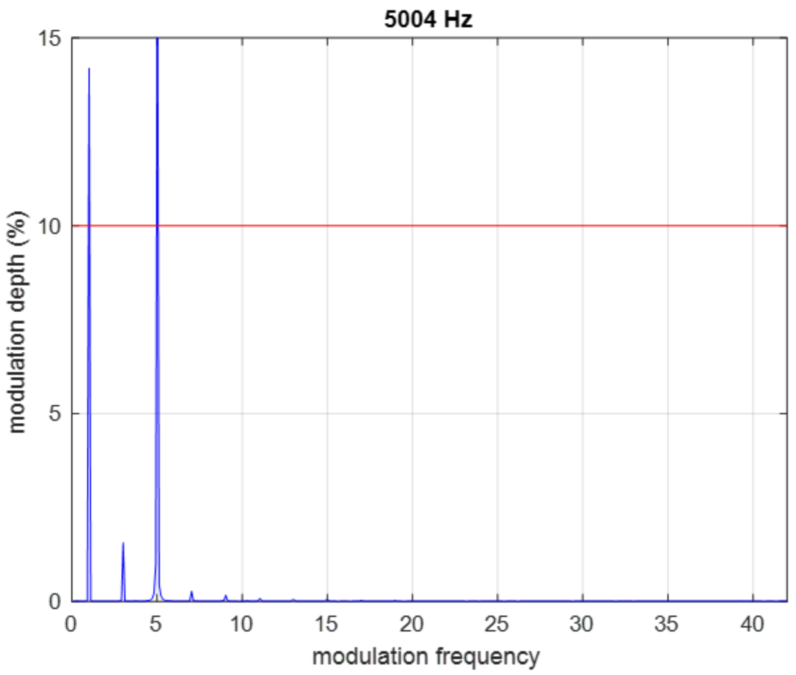


Fig. 3.22.: Modulation information

3.3.1 Detector with Principal Component Analysis

We have already shown that the first stage feature extraction is the output of the detector called the feature matrix in Section 3.2.1. The column dimension of the feature matrix is fixed at 6. But the row dimension of the feature matrix varies for different printer sounds according to the detector algorithm output. As a result, based on this feature matrix, we can further extract a fixed-length feature vector during the intermediate steps in the principal component analysis (PCA) [33, 34] to represent each printer sound.

PCA is an unsupervised linear transformation technique that widely used for feature extraction, data compression, and dimension reduction [35–40]. It performs a linear mapping from original high dimension data to a new lower dimension subspace with the maximum variance in the original one. The orthogonal axes in the new subspace we call them principal components. Although we lose some information due to the reduced dimension, the new feature with lower dimension still retains the most important information because the mapping is based on the maximum variance. Normally, people use PCA as following six steps:

Step 1: Standardize the dataset **X** with dimension $n \times d$, where n is the number of the observations and d is the number of the features.

Step 2: Construct the covariance matrix Σ as equation 3.14 based on equation 3.13

$$\vec{\mu} = \frac{1}{n}\sum_{i=1}^{n}\vec{x}_i \tag{3.13}$$

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}\left(\vec{x}_i - \vec{\mu}\right)^T\left(\vec{x}_i - \vec{\mu}\right) \tag{3.14}$$

where $\vec{x}_i$ is the i-th data.

Step 3: Find all of the eigenvalues, and their corresponding eigenvectors, of the covariance matrix that satisfy the Eigen-equation 3.15.

$$\Sigma\vec{w} = \lambda\vec{w} \tag{3.15}$$

Step 4: Sort the eigenvalues in descending order and its corresponding eigenvectors. The sorted eigenvectors construct a projection matrix $\mathbf{W}$ with dimension $d \times d$. Columns inside the projection matrix are the principal components.

Step 5: Select k eigenvectors which correspond to the k largest eigenvalues ($k \leq d$) to construct a new projection matrix $\mathbf{W}'$.

Step 6: Project the original d-dimensional dataset $\mathbf{X}$ onto the new k-dimensional subspace using the new projection matrix $\mathbf{W}'$.

In our work, we have some changes based on the normal pipeline of the PCA. For each printer sound, detector can extract $n \times 6$ feature matrix where each row is 6-tuple feature vector, and n is the number of feature vectors. The number of n is varied for different printer sound. Each printer sound has to go through the following steps:

Step 1: Based on the extracted feature matrix as shown in Figure 3.23, we normalize each column into unit length with Euclidean norm as shown in Figure 3.24.

$X_{original} =$

11895	23.24793	24.93256	3	0	0
2074	23.12136	29.61779	3	0	0
13148	21.12528	22.71245	3	0	0
14414	15.58967	18.45594	9	1	50.72735
352	13.73516	22.77757	3	0	0

Fig. 3.23.: Example of the feature matrix $\mathbf{X}$ before normalization.

$X_{normalized} =$

0.518376	0.526056	0.465064	0.27735	0	0
0.090383	0.523192	0.552457	0.27735	0	0
0.572981	0.478023	0.423653	0.27735	0	0
0.628152	0.352758	0.344256	0.83205	1	1
0.01534	0.310792	0.424867	0.27735	0	0

Fig. 3.24.: Example of the normalized feature matrix.

Step 2: The same as the normal PCA process. Based on the normalized feature matrix, we can find its mean vector $\vec{\mu}$ and covariance matrix Σ as shown by Equations 3.13 and

3.14, respectively. In our work, $\vec{x}_i$ is the i-th row feature vector within the feature matrix instead of the i-th data.

Step 3: The same as the normal PCA process. Find all of the eigenvalues, and their corresponding eigenvectors, of the covariance matrix that satisfy the Eigen-equation 3.15.

Step 4: The same as the normal PCA process. Sort the eigenvalues in descending order and its corresponding eigenvectors.

Step 5: Finally, we choose the eigenvector that corresponds to the largest eigenvalue, which is also called the first principal component, as our final feature to feed into our anomaly detection model.

All of the printer sounds have to go through the same process from Step 1 to Step 5. Even though the detector extracts feature matrices with different dimensions from different printer sounds, we still can find a feature with fixed dimension to represent each printer sound.

There are two differences between the normal PCA process and our PCA feature. First of all, people normally use PCA on all the acoustic signals at one time. But in our case, we use PCA separately on the feature matrix for each input acoustic signal. Second, people normally use the reduced feature to do further processing. But in our work, we use the first principal component as our feature to do further processing. However, in both cases, we can find the features with fixed dimension to represent each acoustic signal.

Furthermore, the first principal components contains the largest variance from the original data. The way that we calculate how much variance does each principal component contribute is the ratio of the eigenvalue over the sum of them as shown in Equation 3.16.

$$\text{variance} = \frac{\lambda_j}{\sum_{j=1}^{6} \lambda_j} \tag{3.16}$$

Because we sort the eigenvalues in descending order in Step 4, the chosen principal component contains the largest variance of the data.

3.3.2 Mel Frequency Cepstral Coefficients

There are several feature representations for the acoustic signal. For example, image based time-frequency representation of spectrogram, Constant-Q Transform [41], and the most popular spectral based parametric representation of Mel frequency cepstral coefficients [42] (MFCCs). MFCCs is intuitively designed to represent the envelope of the short time power spectrum, which is the shape of the vocal track. Although MFCCs is the dominant feature dealing with speech recognition task [43–45], recently, it is also used in a variety of different tasks that are totally different from speech recognition such as acoustic scene classification [46], environmental sound classification [47], energy management system [48], bridge health monitoring system [49], leak detection of water pipeline [50], and heart rate measurement [51]. The following paragraph will cover the introduction of MFCCs and its extracted features.

We use four terms: Mel frequency, Cepstrum, Mel frequency cepstrum (MFC), and Mel frequency cepstral coefficients (MFCCs) to explain MFCCs. First of all, Mel frequency is a frequency scale that can closely approximate the human auditory system. The transformation between frequency in Hertz (f) and Mel-scale frequency (m) is as shown in Equations 3.17 and 3.18, and Figure 3.25. Note that the frequency below 1000 Hz is approximate a linear transformation.

$$f = 700 \cdot (e^{\frac{m}{1127}} - 1) \tag{3.17}$$

$$m = 1127 \cdot \ln\left(1 + \frac{f}{700}\right) \tag{3.18}$$

Second, the cepstrum [52,53] is the inverse Fourier transform of the log-magnitude Fourier spectrum as shown in Equation 3.19.

$$C = \mathscr{F}^{-1}\{\log(\|\mathscr{F}\{f(t)\}\|^2)\} \tag{3.19}$$

Third, Mel frequency cepstrum (MFC) is a representation of short term power spectrum of linear cosine transform of a log power spectrum on Mel scale frequency. Each short time signal has to go through the following steps: Take Fourier transform, take log of

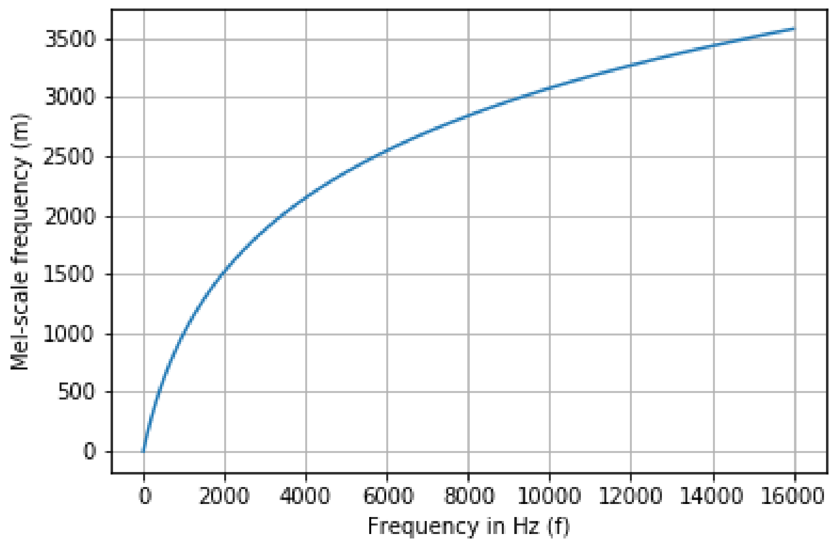


Fig. 3.25.: Non-linear transformation between frequency in Hz (f) and Mel-scale frequency (m).

power, and take discrete cosine transform (DCT). Finally, Mel frequency cepstral coefficients (MFCCs) is a collection of multiple MFCs based on multiple signal framings. Each frame is 20 msec long and the frame step is 10 msec long. In other words, MFCCs is the real cepstrum of a windowed short time signal derived from the FFT of the signal.

The MFCCs pipeline is shown in Figure 3.26. First, segments the input signal $s[n]$ into i frames and then calculate partial Discrete Fourier Transform (DFT) of each frame as shown in Equation 3.20,

$$S_i[k] = \sum_{n=0}^{N-1} s_i[n] e^{-j\frac{2\pi kn}{N}},\ k = 0, \ldots, N-1 \tag{3.20}$$

where N is the length of each frame. Second, calculate power spectral estimation of the i-th frame as shown in Equation 3.21.

$$P_i[k] = \frac{1}{N} |S_i[k]|^2 \tag{3.21}$$

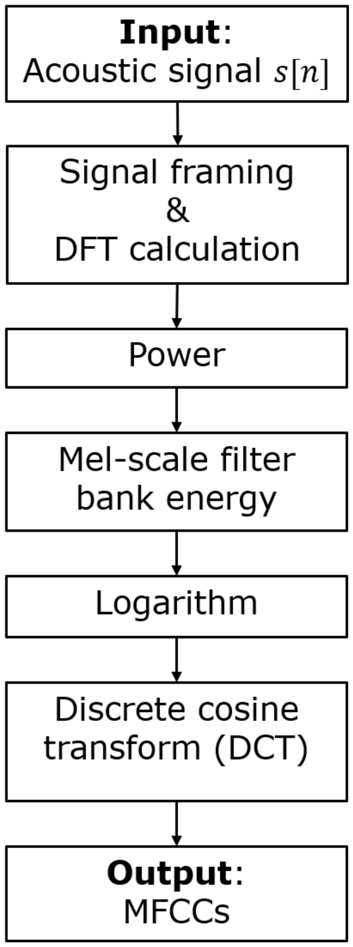

Input:
Acoustic signal $s[n]$
Signal framing & DFT calculation
Power
Mel-scale filter bank energy
Logarithm
Discrete cosine transform (DCT)
Output:
MFCCs

Fig. 3.26.: MFCCs pipeline.

Third, calculate the Mel scale filter bank energy as shown in Equation 3.22,

$$E_i[m] = \sum_{k=0}^{N-1} P_i[k]H_m[k],\ m = 0, ..., M-1 \tag{3.22}$$

where $H_m[k]$ is the m-th Mel filter as shown in Equation 3.23, and Figure 3.27,

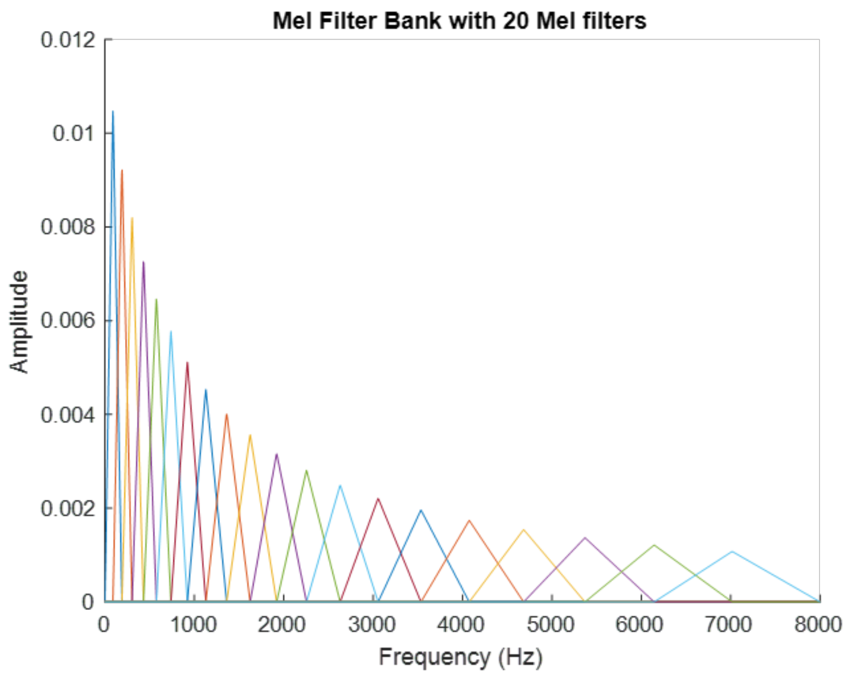


Fig. 3.27.: Example of Mel filter bank with 20 Mel filters.

$$H_m[k] = \begin{cases} 0 & , k < f(m-1) \\ \frac{k - f(m-1)}{f(m) - f(m-1)} & , f(m-1) \leq k < f(m) \\ 1 & , k = f(m) \\ \frac{f(m+1) - k}{f(m+1) - f(m)} & , f(m) < k \leq f(m+1) \\ 0 & , k > f(m+1) \end{cases} \tag{3.23}$$

where M is the number of the Mel filters and $f(\cdot)$ is the list of $M+2$ Mel-spaced frequencies. The filter design satisfies $\sum_{m=0}^{M-1} H_m[k] = 1$. Note that $H_0[k]$ is equal to 0 as shown in Figure 3.28. After calculating the Mel scale filter bank energy, the forth step is to take the log of it as shown in Equation 3.24.

$$E_i'[m] = \log(E_i[m]) \tag{3.24}$$

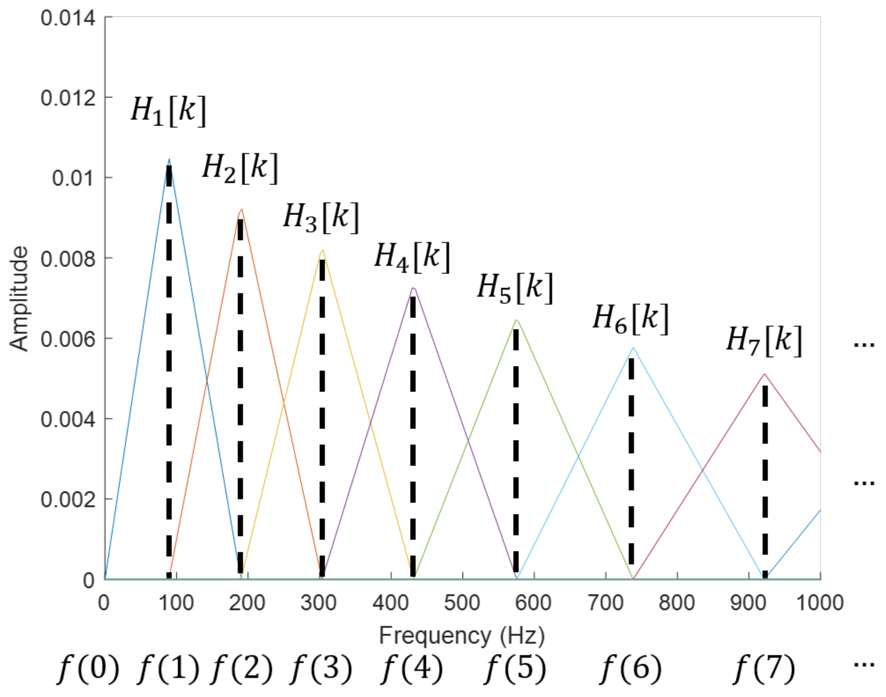


Fig. 3.28.: Explanation of $H_m[k]$ and $f(\cdot)$.

Finally, take the discrete cosine transform (DCT) of $E_i'[m]$ as shown in Equation 3.25.

$$c_i[n] = \sum_{m=0}^{M-1} E_i'[m] \cos\left((m+0.5)\frac{\pi n}{M}\right),\ n = 0, \ldots, M-1 \tag{3.25}$$

where M varies for different implementations. In our, we use the first 10 cepstrum coefficients as the features to represent each audio sample. After taking DCT, we are finished extracting MFCCs feature.

In order to have more mathematical connection with MFCCs, we will explain more about DCT. A discrete cosine tranfsorm [54, 55] (DCT) expresses a finite sequence of data points in terms of a sum of cosine function oscillating at different frequencies as shown in Equation 3.26

$$X[k] = a_k \cdot \sum_{n=0}^{N-1} x[n] \cos\left((n+0.5)\frac{\pi k}{N}\right),\ k = 0, \ldots, N-1 \tag{3.26}$$

where $a_k = \sqrt{\frac{1}{N}}$ when $k = 0$ and $a_k = \sqrt{\frac{2}{N}}$ when $k \neq 0$, and N is the length of signal $x[n]$. We can also say that a DCT is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers. The definition of DFT is shown in Equation 3.27

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi\frac{kn}{N}},\ k = 0, ..., N-1 \tag{3.27}$$

where N is the length of the discrete time signal $x[n]$. In DFT, as shown in Equation 3.27, it has exponential term. On the other hand, in DCT, as shown in Equation 3.26, it only has the cosine term. For an exponential term, it can be written as the combination of a cosine term and a sine term for real and imaginary part, respectively. This is the reason why we can illustrate that the DCT is a Fourier-related transform with only real numbers. There are several reasons why taking DCT at the last step to extract MFCCs. First of all, based on Equation 3.26, the resulting coefficients are real-valued. Second, because the Mel filters are overlapping as shown in Figure 3.27, the filter energies are correlated with each other. The purpose of DCT is to decorrelate the energies between filters. Furthermore, after taking DCT, the filter energies are more robust to noise and spectral estimation errors.

Based on MFCCs, we further demonstrate what features will be extracted as the final features. In some paper, for example, [56] directly take the result of MFCCs as an image to do further processing. However, in our work, we use mean and standard deviation of each coefficient with respect to time as the final feature [57]. Discarding the 0-th coefficient, the first tem MFCCs for each frame are used. Figure 3.29 shows an example of MFCCs with the first 10 coefficients. In Figure 3.30, the vertical axis and horizontal axis represent the order of the coefficient and time, respectively. The bottom row represents the first coefficient with 938 frames across the time, and the top row is the tenth coefficient. For each coefficient, we calculate its mean and standard deviation within these 938 frames with respect to time. The final feature representation based on MFCCs is shown in Figure 3.31. The dimension of it is 1×20. The first 10 values come from means and the following 10 values come from standard deviations for the first 10 MFCCs.

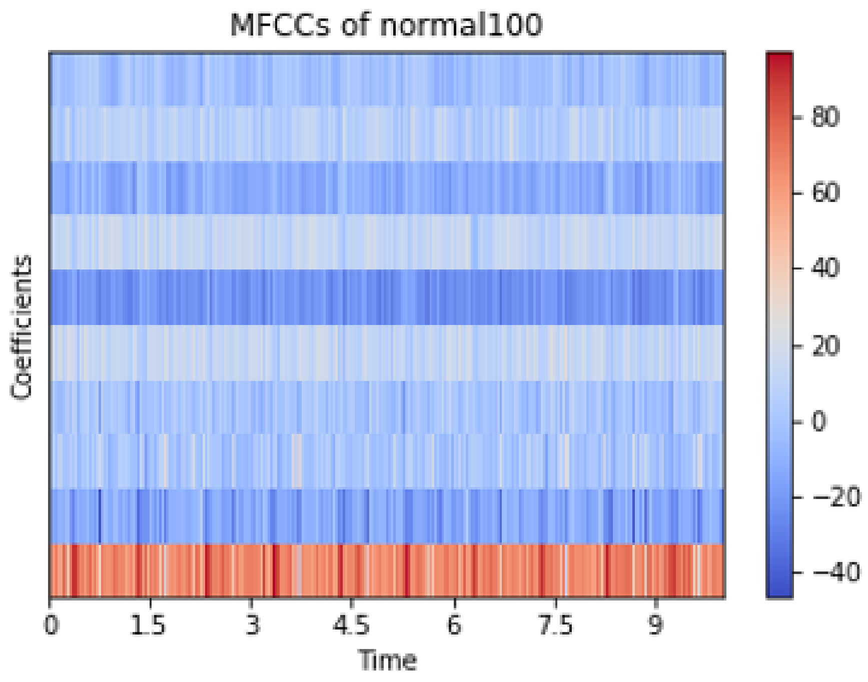


Fig. 3.29.: Example of the first 10 MFCCs.

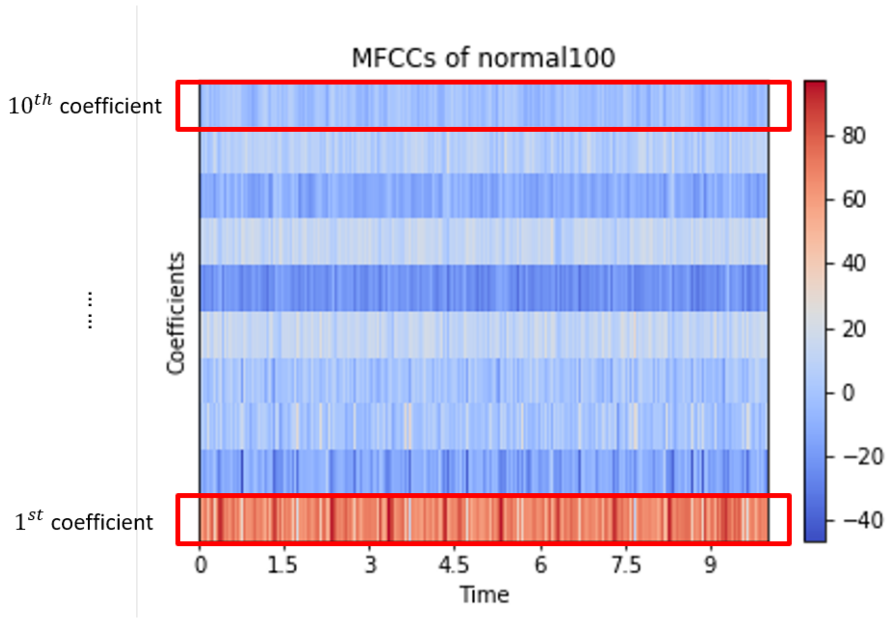


Fig. 3.30.: The order of the coefficients in MFCCs.

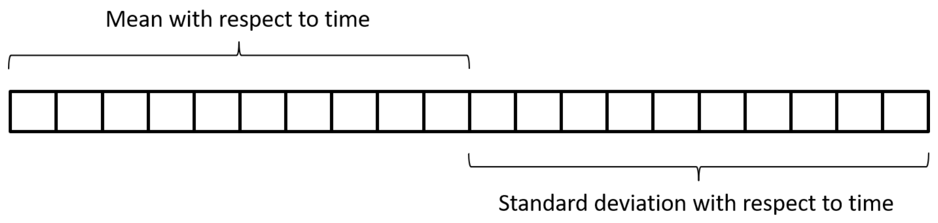

Mean with respect to time
Standard deviation with respect to time

Fig. 3.31.: Final feature representation based on MFCCs.

3.3.3 Detector with Mean and Standard Deviation

As the same reason that we previously mentioned Section 5.3.1, in order to solve the different dimension problem, we would like to extract a feature with fixed dimension to represent each audio file. Based on the output of the detector, we calculate the mean and standard deviation column by column as our final feature representation.

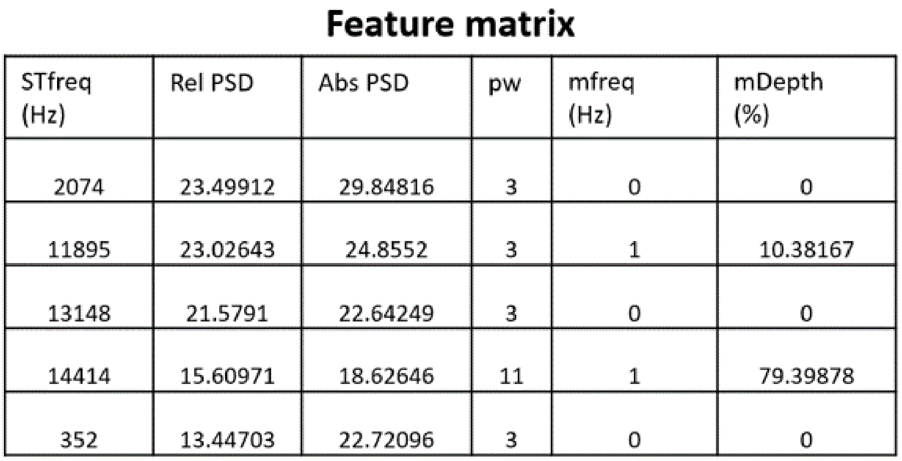
For each audio file, the detector can extract an $n \times 6$ feature matrix where each row is 6-tuple feature vector, and n is the number of feature vectors. The number of n is varied for different audio file. Each audio file has to go through the following steps:

Step 1: Based on the extracted feature matrix as shown in Figure 3.32, we normalize each column into unit length with Euclidean norm as shown in Figure 3.33.

Step 2: Calculate the mean and standard deviation column by column. For each column in the feature matrix, we use two values, the mean and standard deviation, as the features to represent each extracted characteristic.

Step 3: Finally, we concatenate all calculated means and standard deviations into a one-row feature vector as the final feature representation to represent each audio file as shown in Figure 3.34

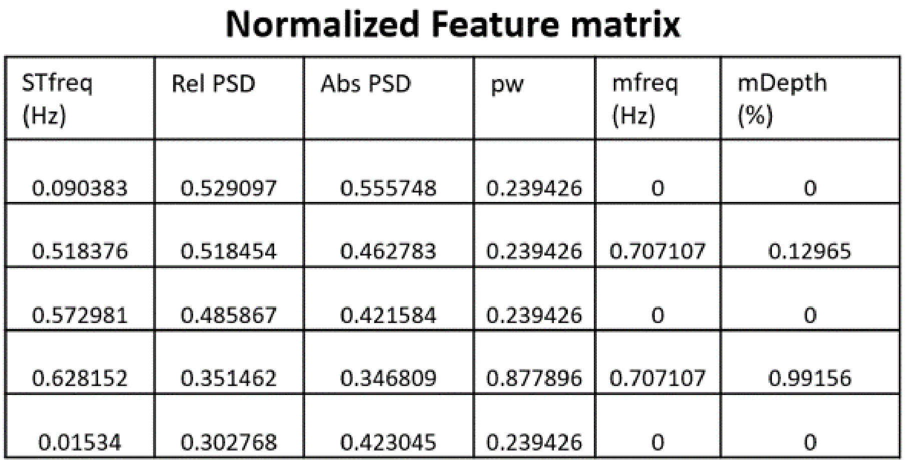
The detector with mean and standard deviation is inspired by MFCCs in Section 3.3.2. The difference between Section 2.3.2 and 2.3.3 is that the feature introduced in Section 2.3.2 calculates the mean and standard deviation with respect to time. On the other hand,



Feature matrix

STfreq (Hz)	Rel PSD	Abs PSD	pw	mfreq (Hz)	mDepth (%)
2074	23.49912	29.84816	3	0	0
11895	23.02643	24.8552	3	1	10.38167
13148	21.5791	22.64249	3	0	0
14414	15.60971	18.62646	11	1	79.39878
352	13.44703	22.72096	3	0	0

Fig. 3.32.: Example of the feature matrix. From the first to the last columns are: Strong tone frequency, relative PSD, absolute PSD, peak width, modulation frequency, and modulation depth, respectively.



Normalized Feature matrix

STfreq (Hz)	Rel PSD	Abs PSD	pw	mfreq (Hz)	mDepth (%)
0.090383	0.529097	0.555748	0.239426	0	0
0.518376	0.518454	0.462783	0.239426	0.707107	0.12965
0.572981	0.485867	0.421584	0.239426	0	0
0.628152	0.351462	0.346809	0.877896	0.707107	0.99156
0.01534	0.302768	0.423045	0.239426	0	0

Fig. 3.33.: Normalized feature matrix.

the feature illustrate in Section 2.3.3 calculates the mean and standard deviation with respect to different extracted characteristics based on the detector.

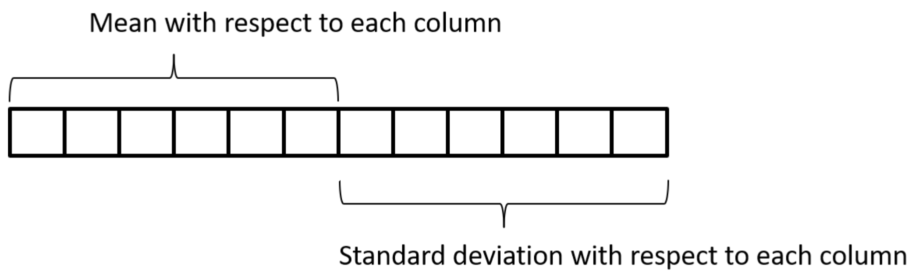

Mean with respect to each column
Standard deviation with respect to each column

Fig. 3.34.: Final feature representation based on detector with mean and standard deviation.

4. ANOMALY DETECTION MODELS AND DATA AUGMENTATION METHODS

4.1 Classifier

We use the semi-supervised classifier one class support vector machine (OCSVM) [58] and the supervised classifiers support vector machine (SVM) [59–61], random forest (RF) [62], and *k*-nearest neighbor (*k*-NN) [63, 64] as our classifiers.

4.1.1 One Class Support Vector Machine

Semantically, one class support vector machine (OCSVM) only learns for one class. Corresponding to our case, which is anomaly detection (or outlier detection), we can use OCSVM as our classifier to classify the input audio as normal or not normal. The reason why we are using OCSVM is that it only needs the training data for one class, which is suitable for our situation that we only have collected normal printer sounds. The abnormal sounds are synthesized by the defect generator. Because OCSVM only learns for one class, practically, we train OCSVM with real normal printer sounds and the augmented normal printer sounds separately. For testing, we use the real normal and the synthetic abnormal printer sounds as our testing data. The testing data will be the same for different training dataset.

The concept of OCSVM [58] is that it maps input data into a high dimensional feature space via a kernel and finds the maximal margin hyperplane which best separates the training data from the origin in the mapped feature space. In mathematical terms, OCSVM is solving the optimization problem stated in Equation 4.1 [58]

$$
\begin{aligned}
&\min_{\mathbf{w},\xi_i,\rho} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{vl}\sum_{i=1}^{l}\xi_i - \rho \\
&\text{subject to} \left(\mathbf{w}^T\Phi(\mathbf{x}_i)\right) \leq \rho - \xi_i,\ i = 1,...,l,\ \xi_i \geq 0
\end{aligned}
\tag{4.1}
$$

where

$$
\begin{aligned}
\mathbf{w} &= \sum_i \alpha_i \cdot \Phi(\mathbf{x}_i) \\
\sum_i \alpha_i &= 1
\end{aligned}
\tag{4.2}
$$

The decision function is

$$
f(\mathbf{x}) = sign\left(\mathbf{w}^T\Phi(\mathbf{x}_i) - \rho\right) \tag{4.3}
$$

where $v \in (0,1]$ is the fraction of outliers, l is the number of data points, ξ_i is the slack variable, ρ is an offset parameter associated with the kernel, and $\Phi(\cdot)$ is the kernel function that maps the training sample into another space. The way we categorize the data as normal or abnormal is based on the decision function. If $f(\mathbf{x}) > 0$, we label $\mathbf{x}$ as normal, and if $f(\mathbf{x}) < 0$, we label $\mathbf{x}$ as abnormal.

4.1.2 Support Vector Machine

Support vector machine (SVM) [60] is a supervised classifier. The concept of it is similar to OCSVM. The difference between SVM and OCSVM is that OCSVM separates the training data from the origin in the mapped feature space. One the other hand, SVM separates the multi-class training data in the mapped feature space. In mathematical terms, SVM is solving the optimization problem stated in Equation 4.4 [60]

$$
\begin{aligned}
&\min_{\mathbf{w},\xi_i,b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i \\
&\text{subject to} \left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right) \leq 1 - \xi_i,\ i = 1,...,l,\ \xi_i \geq 0
\end{aligned}
\tag{4.4}
$$

where

$$\begin{aligned} \mathbf{w} &= \sum_i \alpha_i \cdot \Phi(\mathbf{x}_i) \\ \sum_i \alpha_i &= 1 \end{aligned} \tag{4.5}$$

The decision function is

$$f(\mathbf{x}) = \operatorname{sign}\left(\sum_{i=1}^{l} y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) + \rho\right) \tag{4.6}$$

where C is a trade-off of misclassification of training samples against simplicity of the separation hyperplane, b is offset of the hyperplane, and y_i is the label of training data x_i. Also, as the same description in OCSVM, l is the number of data points, $K(\mathbf{x}_i, \mathbf{x}) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x})$ is the kernel, $\phi(\cdot)$ is the function that maps the training sample into higher dimensional space, and ξ_i is the slack variable. The way we categorize the data as normal or abnormal is based on the decision function. If $f(\mathbf{x}) > 0$, we label $\mathbf{x}$ as normal, and if $f(\mathbf{x}) < 0$, we label $\mathbf{x}$ as abnormal.

If the data are linearly separable, then a linear kernel as shown in Equation 4.7 will work well. However, if the data is not linearly separable, then a non-linear kernel, namely the radial basis function (RBF) as shown in Equation 4.8, should be used. 4.8.

$$K(\mathbf{x}_i, \mathbf{x}) = \mathbf{x}_i^T \mathbf{x} \tag{4.7}$$

$$K(\mathbf{x}_i, \mathbf{x}) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}\|^2} \tag{4.8}$$

where $\mathbf{x}_i, \mathbf{x} \in \mathbb{R}^d$, and d is the number of the feature dimension.

4.1.3 Random Forest

Random forest (RF) [62] is a supervised ensemble classifier, which is constructed by a set of multiple decision trees. The more trees it has, the more robust the forest is. Each decision tree will report class prediction. Finally, we follow the wisdom of crowds rule to classify the input to the class with the most votes. Take Figure 4.1 as an example, if the forest has three decision trees, one of them predicts the input as class 1 and remaining two of them predict the input as class 0. Based on the number of votes, we will categorized the input as class 0.

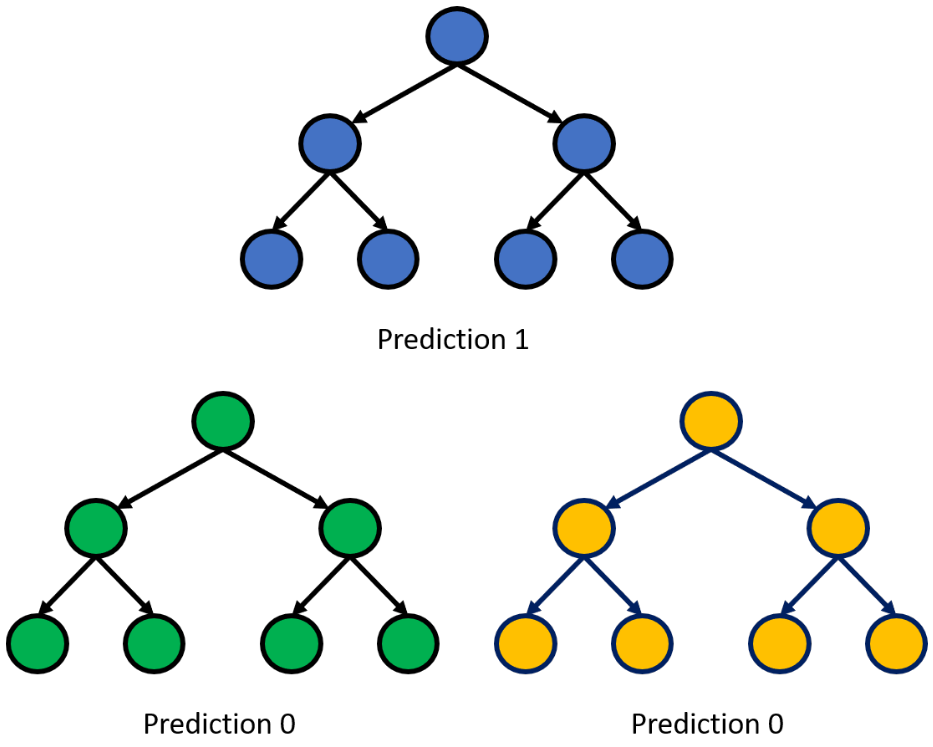


Fig. 4.1.: Voting for multiple decision trees

Combination of multiple decision trees is not the only reason for why random forest is powerful. Bagging, a method to generate a new training dataset by randomly choosing the feature from the original dataset with replacement, is another reason.

Bagging (Bootstrap Aggregation) [62] is not extracting the subset from the original dataset but the subset of the feature. Most important of all, even if we choose the subset of the feature, the dimension of the subset of the feature still remains the same. For example, if we have a data with dimension $N = 4$ as [1,2,3,4], the new training data still has the same dimension of $N = 4$ but is constructed by the randomly selected feature with replacement such as [1,1,3,3]. The dimension of the data is not changed but there are some randomly selected features repeated with replacement. In RF, each individual decision tree randomly picks the subset of the features to decrease the correlation between trees.

4.1.4 K-Nearest Neighbor

The k-nearest neighbor (k-NN) [63] is a supervised classifier based on the similarity measure to classify the data with the majority vote among the k-nearest neighbor. First of all, calculate the distance between the testing sample with all of the training samples. Second, sort the calculated distances in ascending order and the corresponding label of the training data. Third, count the number of the label of each class from the first k sorted distances. Finally, the testing sample will be classified to the class with the largest count. In this dissertation, we use Euclidean distance as our similarity measurement method as shown in Equation 4.9 to calculate the distance between the training samples and testing sample.

$$\text{Euclidean distance function} = \sqrt{\sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{x})^2} \tag{4.9}$$

where $\mathbf{x}_i$ is the i-th training sample, $\mathbf{x}$ is the testing sample, and N is the number of the training samples. For example, if $k = 1$, then the testing sample will be categorized to the class with the shortest distance.

4.2 Data Augmentation

There are three reasons for us to do data augmentation. First of all, different tasks need different data. Typically, people are solving the problems such as ASR and ASC; and currently, there is no suitable public dataset that we can use as a comparison baseline. Second reason is the lack of the acoustic data. We utilize data augmentation to increase the number of datasets. The third reason is to decrease feature development cycle. We use the following six augmentation methods to generate ten augmented datasets.

The following methods are a brief overview of acoustic data augmentation methods based on different feature. [65] transforms the spectrogram using a random linear warping along the frequency dimensions. [66] uses the modified version of [65] with a fixed gap of warping factor. [67] proposes Equalized Mixture Data Augmentation (EMDA) to augment the sound by randomly mixing two sounds of a class, with randomly selected timings. Furthermore, this method perturbs the sound by amplifying/attenuating a particular frequency band. Similarly, [68] makes the assumption that a combination of two or more audio segments from the same scene is another sample of that scene with more complex pattern end events. Also, [69, 70] mixes training samples together to do augmentation. Instead of mixing audio sequences, [71] inserts blank rectangles into the two-dimensional Mel-spectrogram at a randomly chosen size and location to remove some information as their augmentation method.

4.2.1 Pitch Shifting

For pitch shifting [72], we can choose higher or lower pitch of the audio sample. Take Figure 4.2 as an example. If your basic tone of the audio is F, and you tune it from F to $F^{\#}$, this process is called tuning audio file into higher pitch with one musical semitone. On the other hand, if the basic tone of the audio is $F^{\#}$, and you tune it from $F^{\#}$ to F, this process is called tuning audio file into lower pitch with one musical semitone. More specifically, in order to increase the pitch by n musical semitones, we will multiply the frequency by a factor of $2^{\frac{n}{12}}$. For example, if we double the frequency of the audio sample, then it will increase the pitch of 12 semitones. For larger and smaller pitch shifting, we pitch shift with values ±1, ±2 and ±0.1, ±0.2, respectively. Plus sign means higher pitch and minus sign means lower pitch.

Figure 4.3 shows the information of the real printer sound as a comparison baseline. The top two parts are the PSD, strong tone frequencies, and modulation frequencies information, and the bottom table is the feature matrix. Figures 4.4, 4.5, and 4.6 show the information of its augmented printer sounds with different pitch shifting parameters. Figures 4.4 and 4.5 illustrate the difference between larger and smaller pitch shifting, and Figures 4.4 and 4.6 illustrate the difference between higher and lower pitch shifting, respectively.

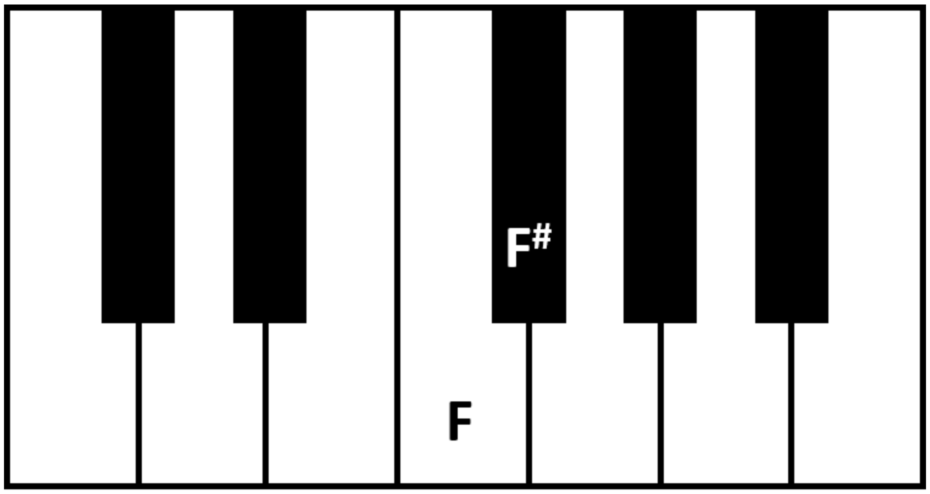
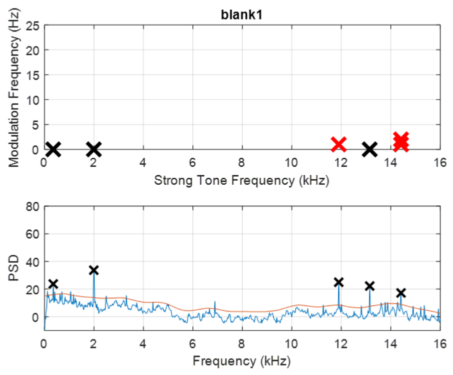
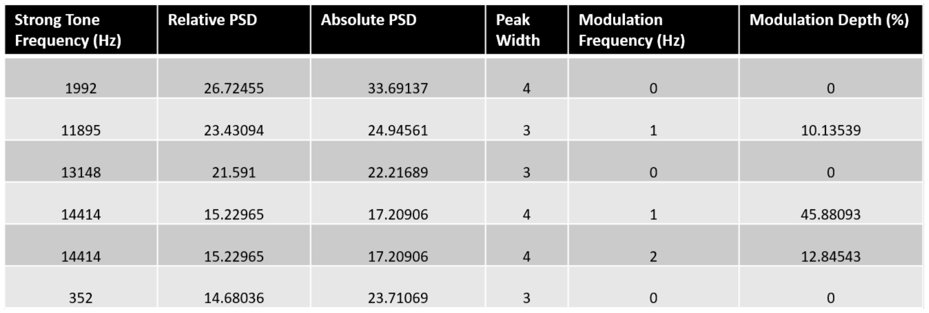


Fig. 4.2.: Example of a piano.

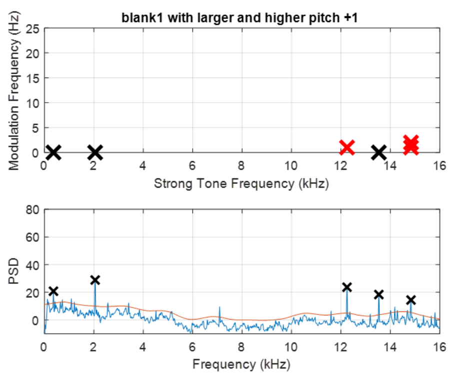




Strong Tone Frequency (Hz)	Relative PSD	Absolute PSD	Peak Width	Modulation Frequency (Hz)	Modulation Depth (%)
1992	26.72455	33.69137	4	0	0
11895	23.43094	24.94561	3	1	10.13539
13148	21.591	22.21689	3	0	0
14414	15.22965	17.20906	4	1	45.88093
14414	15.22965	17.20906	4	2	12.84543
352	14.68036	23.71069	3	0	0

Fig. 4.3.: Extracted features from a real printer sound.

For both Figures 4.4 and 4.5, their PSD will shift to right compared with Figure 4.3. But with larger pitch shifting as shown in Figure 4.4, the PSD will shift to right more than with smaller pitch shifting as shown in Figure 4.5. As a result, the extracted strong tone



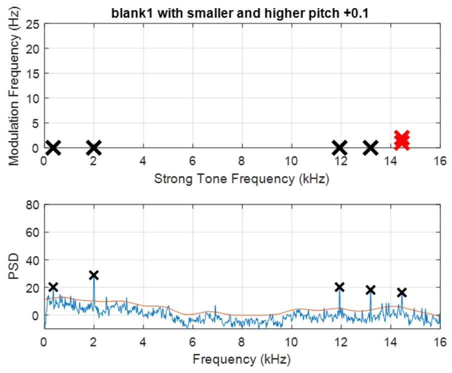
Strong Tone Frequency (Hz)	Relative PSD	Absolute PSD	Peak Width	Modulation Frequency (Hz)	Modulation Depth (%)
2051	25.80075	28.88793	4	0	0
12246	25.63686	23.68972	3	1	11.37195
13535	21.57925	18.37757	3	0	0
14836	15.99574	14.43045	4	1	47.92675
14836	15.99574	14.43045	4	2	12.44941
363	15.52885	20.76689	3	0	0

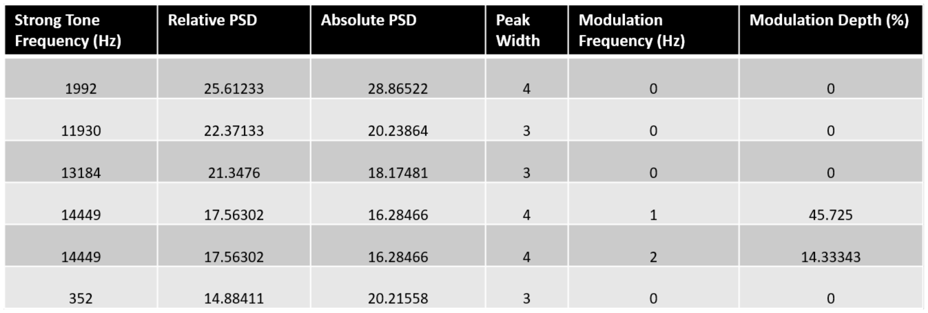
Fig. 4.4.: Extracted features with higher and larger pitch shifting.

frequencies with larger pitch shifting are higher than with smaller pitch shifting. With higher pitch shifting as shown in Figure 4.4, the PSD would shift to right, and the PSD would shift to left with lower pitch shifting as shown in Figure 4.6. As a result, higher pitch shifting will extract higher strong tone frequencies and lower pitch shifting will extract lower strong tone frequencies. Based on the mentioned characteristics, pitch shifting augmentation method will mainly affects the extracted strong tone frequencies.

4.2.2 Time Stretching

For time stretching [72], we can choose to speed it up or slow it down. We stretch with values ±0.01, ±0.02. Plus sign means faster speed ratio and minus sign means slower speed ratio. Figure 4.3 is a comparison baseline, and Figures 4.7 and 4.8 show the feature with faster and slower speed ratio, respectively. With faster speed ratio as shown in Figure 4.7, the extracted modulation frequencies will increase. On the other hand, the extracted





Strong Tone Frequency (Hz)	Relative PSD	Absolute PSD	Peak Width	Modulation Frequency (Hz)	Modulation Depth (%)
1992	25.61233	28.86522	4	0	0
11930	22.37133	20.23864	3	0	0
13184	21.3476	18.17481	3	0	0
14449	17.56302	16.28466	4	1	45.725
14449	17.56302	16.28466	4	2	14.33343
352	14.88411	20.21558	3	0	0

Fig. 4.5.: Extracted features with higher and smaller pitch shifting.

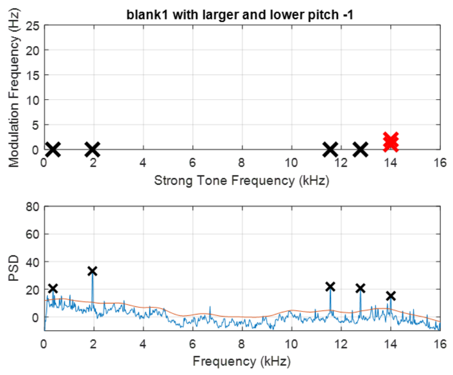
modulation frequencies will decrease with slower speed ratio as shown in Figure 4.8. Based on the mentioned characteristics, time stretching augmentation method will mainly affects the extracted modulation frequencies.

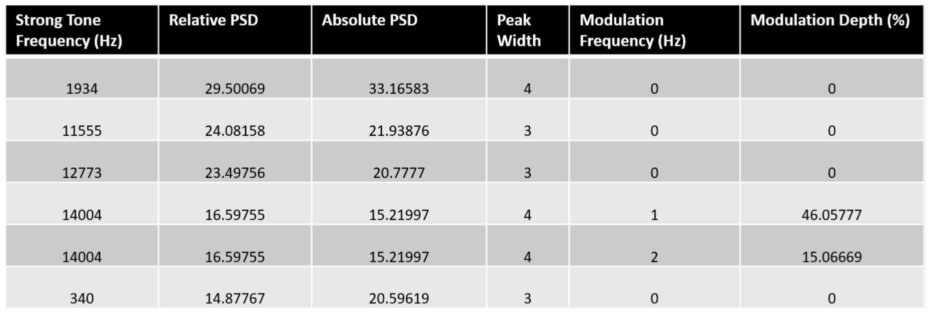
4.2.3 Mixture

For mixture, we follow the similar concept and assumption from [67–70,73–75]. Three different types of mixture are used: Average mixture, alpha mixture, and filter bank alpha mixture.

For average mixture, first, we randomly choose two audio files from the real printer sound dataset. Next,in the time domain, we combine them as shown in Equation 4.10.

$$x_n = 0.5x_i + 0.5x_j \tag{4.10}$$





Strong Tone Frequency (Hz)	Relative PSD	Absolute PSD	Peak Width	Modulation Frequency (Hz)	Modulation Depth (%)
1934	29.50069	33.16583	4	0	0
11555	24.08158	21.93876	3	0	0
12773	23.49756	20.7777	3	0	0
14004	16.59755	15.21997	4	1	46.05777
14004	16.59755	15.21997	4	2	15.06669
340	14.87767	20.59619	3	0	0

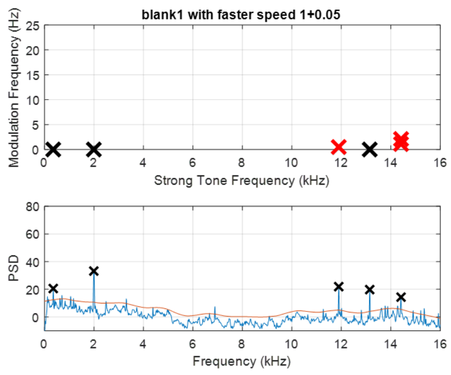
Fig. 4.6.: Extracted features with lower and larger pitch shifting.

where x_n is the augmented mixture result and x_i and x_j are the audio files from the same class. The augmented result keeps the same label as its mixture source.

For alpha mixture, we follow the similar concept of the mentioned average mixture, but change its scaling factor from 0.5 to α. First of all, we randomly choose two audio files from the real printer sound dataset. Next, in the time domain, we combine them with randomly selected scaling factor α as shown in Equation 4.11

$$x_n = \alpha \cdot x_i + (1-\alpha) \cdot x_j \tag{4.11}$$

where $\alpha \in (0,1)$, x_n is the augmented alpha mixture result and x_i and x_j are the audio files from the same class. We can see that if α is equal to half, then Equation 4.11 will be the same as Equation 4.10. We can also say that average mixture is a special case in alpha mixture.

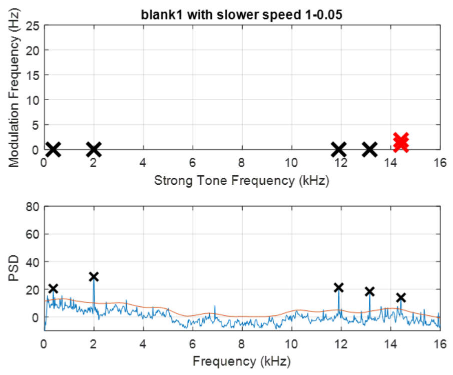


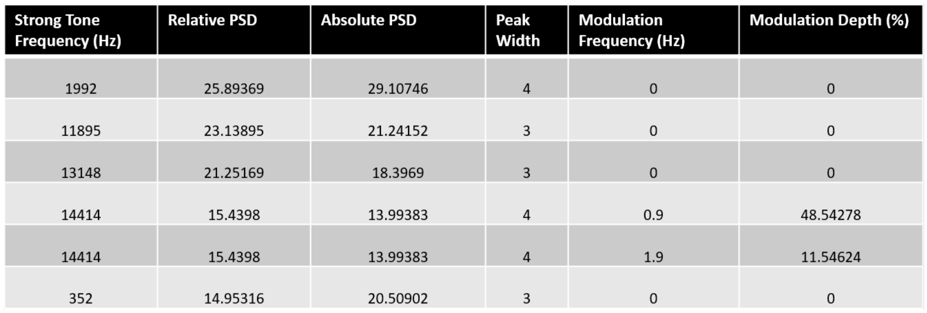
Strong Tone Frequency (Hz)	Relative PSD	Absolute PSD	Peak Width	Modulation Frequency (Hz)	Modulation Depth (%)
1992	29.51821	33.18261	5	0	0
11895	23.94133	21.91565	3	0.5	22.45198
13148	22.72064	19.78416	3	0	0
14414	15.80846	14.34151	4	1.1	47.12084
14414	15.80846	14.34151	4	2.1	12.40086
352	15.10619	20.57205	3	0	0

Fig. 4.7.: Extracted features with faster speed ratio.

For filter bank alpha mixture, we first mention the concept of filter bank reconstruction, and then further illustrate filter bank alpha mixture augmentation method. Figure 4.9 shows the pipeline of filter bank reconstruction. The purpose of filter bank reconstruct is to let output $y[n]$ is equal to input $s[n]$. As shown in Figure 4.9, we use two channel filter banks, low pass filter and high pass filter, as an example, and their amplitude of the frequency response are shown in Figure 4.10. The process of filter bank reconstruction is: First, design the filters. Second, convolve the input signal $s[n]$ with the designed filters $h_0[n]$ and $h_1[n]$. The middle outputs would be $y_0[n]$ and $y_1[n]$, respectively. Finally, sum all of the middle outputs up to obtain the output $y[n]$. Note that the purpose of data augmentation is to augment the similar sample, not the same one. As a result, we add some scaling factors at the middle outputs as shown in Figure 4.11 and Equation 4.12.

$$y_n = (1+\alpha)\cdot y_0 + (1-\alpha)\cdot y_1 \tag{4.12}$$





Strong Tone Frequency (Hz)	Relative PSD	Absolute PSD	Peak Width	Modulation Frequency (Hz)	Modulation Depth (%)
1992	25.89369	29.10746	4	0	0
11895	23.13895	21.24152	3	0	0
13148	21.25169	18.3969	3	0	0
14414	15.4398	13.99383	4	0.9	48.54278
14414	15.4398	13.99383	4	1.9	11.54624
352	14.95316	20.50902	3	0	0

Fig. 4.8.: Extracted features with slower speed ratio.

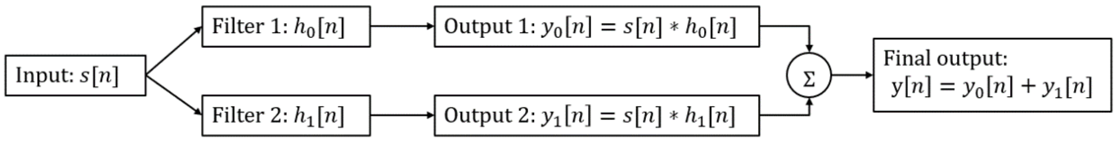

Input: $s[n]$
Filter 1: $h_0[n]$
Filter 2: $h_1[n]$
Output 1: $y_0[n] = s[n] * h_0[n]$
Output 2: $y_1[n] = s[n] * h_1[n]$
Σ
Final output: $y[n] = y_0[n] + y_1[n]$

Fig. 4.9.: Pipeline of filter bank reconstruction.

where α belongs to normal distribution with $\mu = 0$ and $\sigma = 0.1$, and excludes 0.

4.2.4 Concatenation

For concatenation, we have three steps. First of all, randomly choose two audio samples from the real printer sound dataset with the same class. Next, randomly crop five seconds time period from the chosen audio samples as shown in Figure 4.12. Finally, concatenate two cropped audio samples into a ten-second augmented audio sample as shown in Figure 4.13. The reason to choose five seconds is based on printer's print rate and the continuity.

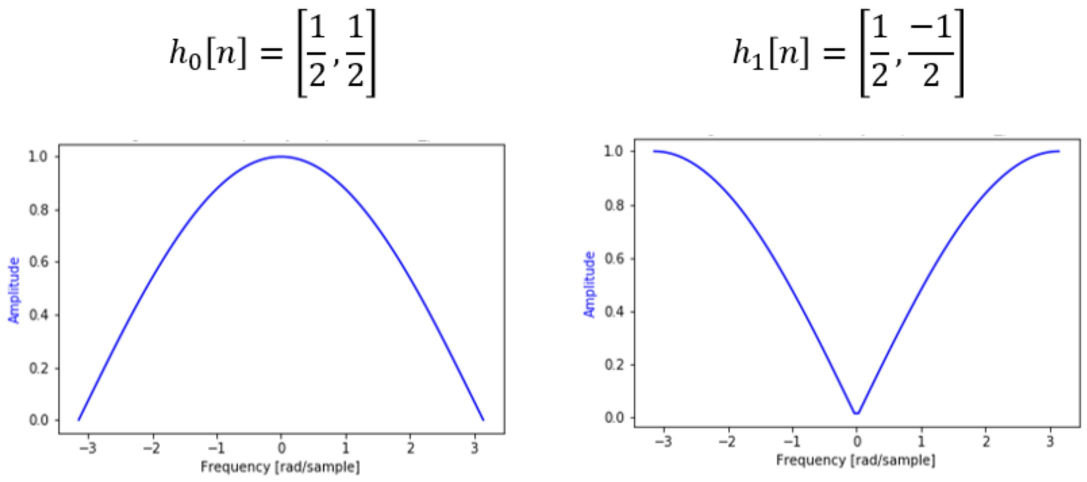


Fig. 4.10.: Frequency response of low pass filter on the left and high pass filter on the right

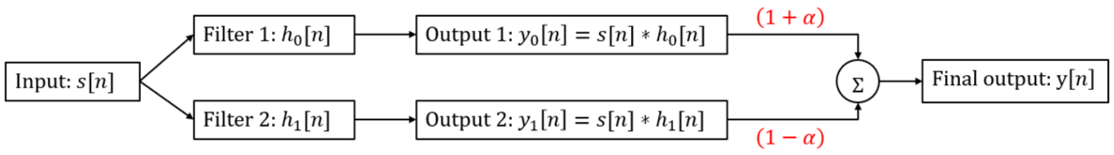


Fig. 4.11.: Pipeline of filter bank alpha mixture.

The print rate of the printer is 60 pages per minute (ppm), which means the shortest time period that we can crop without loosing the continuity is one second. In order to keep the high similarity between the real and augmented audio sample, we decide the cropped time period is five seconds.

4.2.5 Random Erasing

For random erasing, we follow the core concept in [71], which proposes an image data augmentation method. In [71], they randomly select a rectangle region in an image and erases its pixels with random values. We utilize this idea for the image feature representation of the audio sample: Spectrogram. Spectrogram is a 3D information as shown in

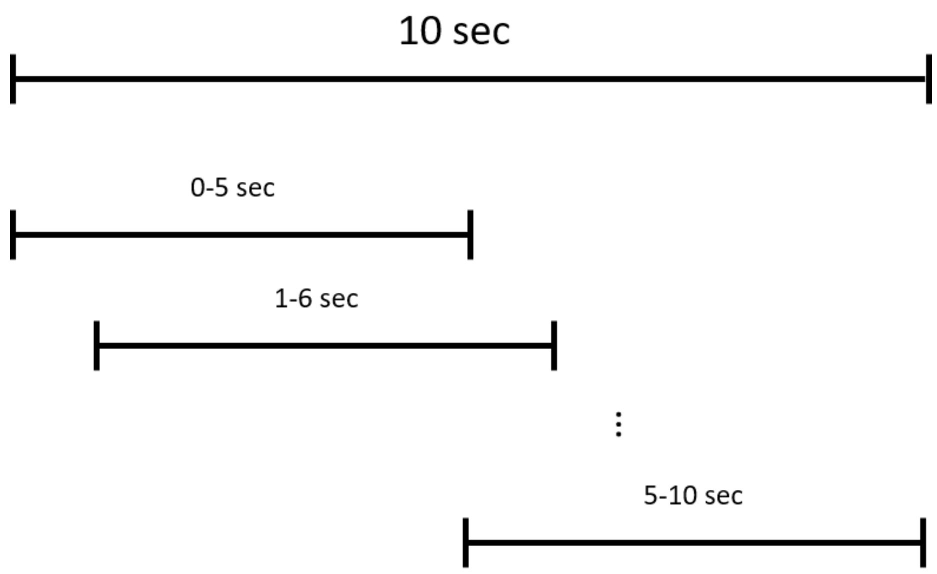


Fig. 4.12.: The second step for concatenation: Randomly crop five seconds time period of the chosen audio sample.

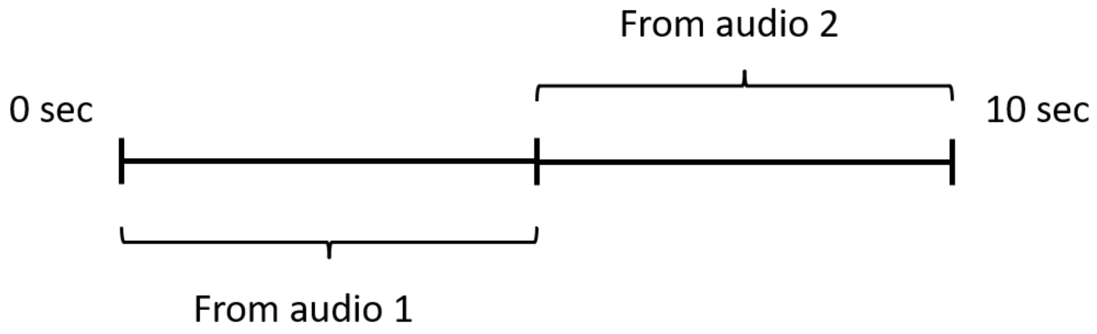


Fig. 4.13.: The last step for concatenation: Concatenate two cropped audio samples into a ten-second audio sample.

Figure 4.14. Vertical axis and horizontal axis represent frequency in hertz and time in second, respectively. Furthermore, the decibel value represents the amplitude of the audio sample in certain frequency band and at certain time. More specifically, instead of spectrogram, we use the result which is one step before we obtain the spectrogram: The short time Fourier transform (STFT) complex-valued matrix. As shown in Figure 4.14, spectrogram is the result after taking the absolute value of the STFT complex-valued matrix. Based

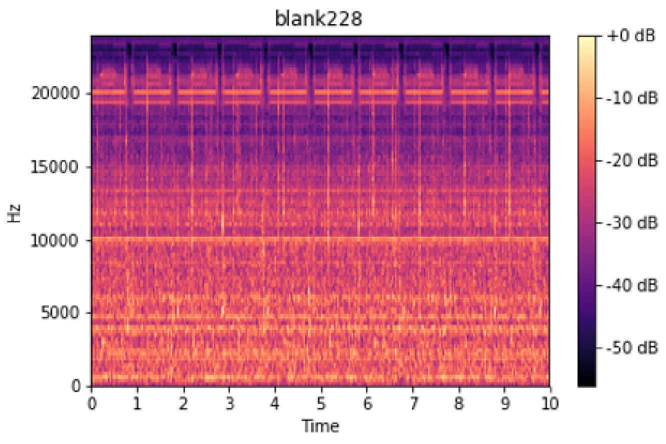


Fig. 4.14.: Example of spectrogram.

on the STFT complex-valued matrix, we randomly select the size and the position of the rectangle, and replace all pixels inside the rectangle region with the random value ranges from 0 to 1 for both real and imaginary parts as shown in Figure 4.15. Note that the size

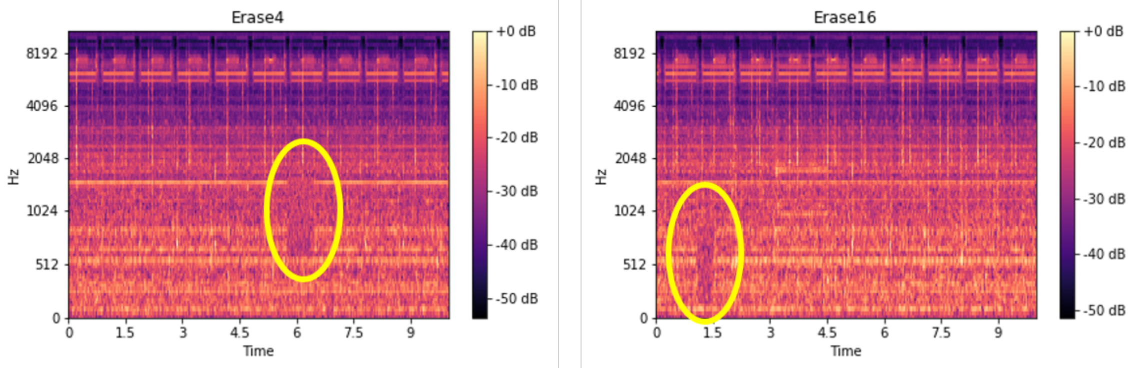
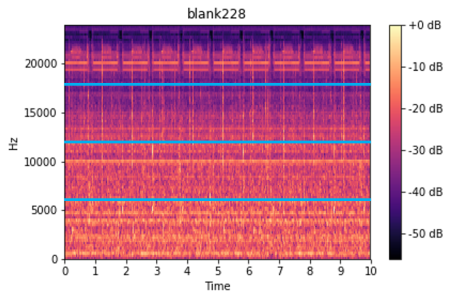


Fig. 4.15.: Examples for random erasing augmentation method. The yellow circle part is the erasing part.

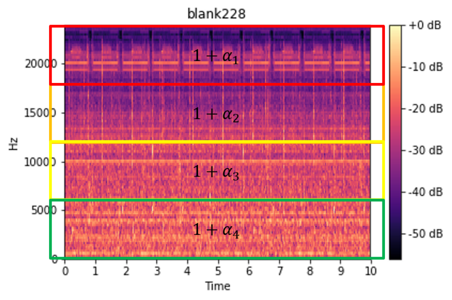
of STFT complex-valued matrix is 1024×938, and the width and the height of the erasing rectangle is randomly chosen from 30 to 70.

4.2.6 Zoning Blending

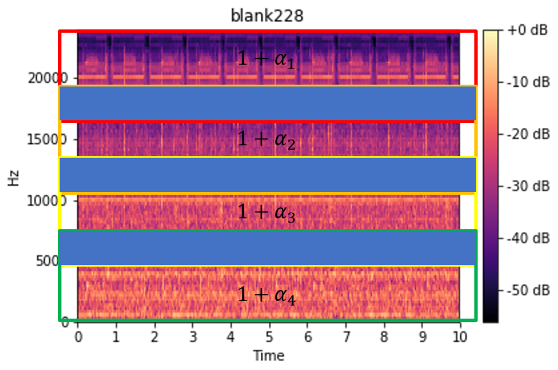
For zoning blending, the zoning concept is inspired by [41,76,77]. In these papers, they propose a zoning technique to obtain the local information by dividing the time frequency representation (TFR) into n zones horizontally as shown in Figure 4.16(a). In this example, the number of n is equal to 4. We go through the following four steps as shown in Figure 4.16 to do zoning blending. First, cut the STFT complex-valued matrix horizontally into four zones as shown in Figure 4.16(a). The top zone is called zone1 and the bottom zone is called zone4. Note that we use spectrogram as an easier representation of STFT complex-valued matrix. Next, randomly select four scaling factors: $1+\alpha_1$, $1+\alpha_2$, $1+\alpha_3$, and $1+\alpha_4$, where $\alpha_i, i = 1, ..., 4$ belong to normal distribution with $\mu = 0$ and $\sigma = 0.1$, and excludes 0 as shown in Figure 4.16(b). But in Figure 4.16(b), there may cause a sharp transition between different zones. As a result, in the third step, center at the zone boundary, we extend the zone boundary from a line to a rectangle region as shown in Figure 4.16(c). The last step for zoning blending is shown in Figure 4.16(d), the left hand side shows our blending rule. For non-blending region, we multiply the fixed scaling factor as we mentioned previously. For blending region, take the blue point as an example, because the position is between zone1 and zone2 with scaling factors $1+\alpha_1$ and $1+\alpha_2$, respectively, the scaling factor at this frequency band is equal to $\frac{1}{2}(1+\alpha_1)+\frac{1}{2}(1+\alpha_2)$. If the blue point towards the higher frequency band, zone1, then the scaling factor for the blue point will contain higher ratio from $1+\alpha_1$ and less ratio from $1+\alpha_2$. In this case, we can avoid the sharp transition between different zones.



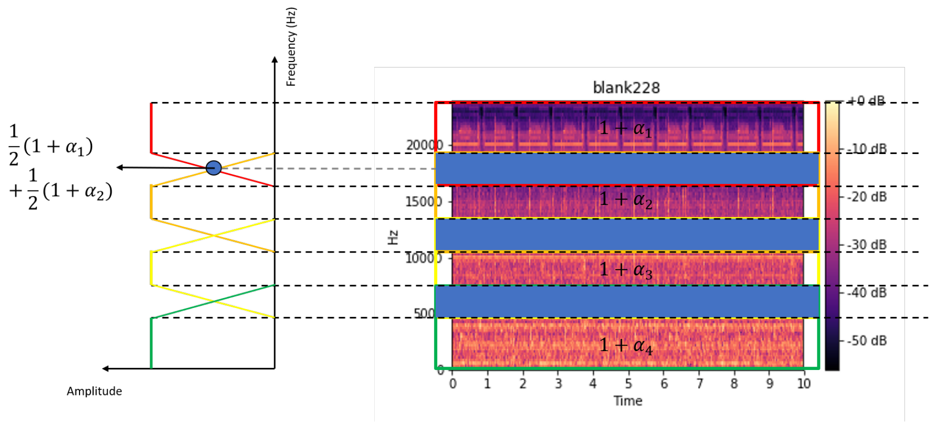
(a)



(b)



(c)



(d)

Fig. 4.16.: The procedures of zoning blending (a) Cut four zones horizontally (b) Randomly select four scaling factors (c) Annotate blending region (d) The last step for zoning blending.

5. DATA ARRANGEMENT AND EXPERIMENTAL RESULTS

5.1 Data Arrangement

We use our collected 400 real normal printer sounds dataset as a comparison basedline. We additionally synthesize 400 abnormal printer sounds as mentioned in Section 3.2.3. Among these 400 synthetic abnormal printer sounds, all of them are used for RF and 40 of them are used for OCSVM. These 400 collected normal plus 40 synthesized abnormal printer sounds are the data arrangement for OCSVM and 400 collected normal plus 400 synthesized abnormal printer sounds are the data arrangement for RF.

For semi-supervised classifier, OCSVM, we randomly choose 360 within the 400 real normal printer sounds as the normal training data as shown in Figure 5.1. The remaining 40 real normal printer sounds and the 40 synthetic abnormal printer sounds are the testing data as shown in Figure 5.2.

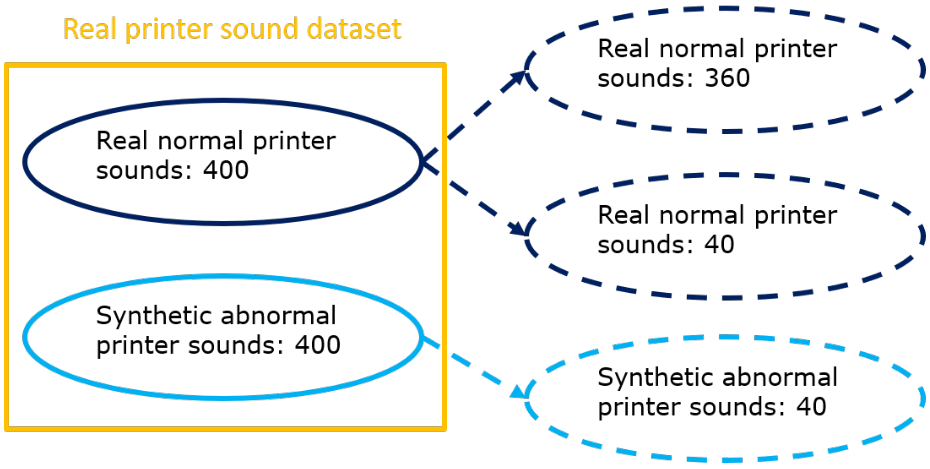
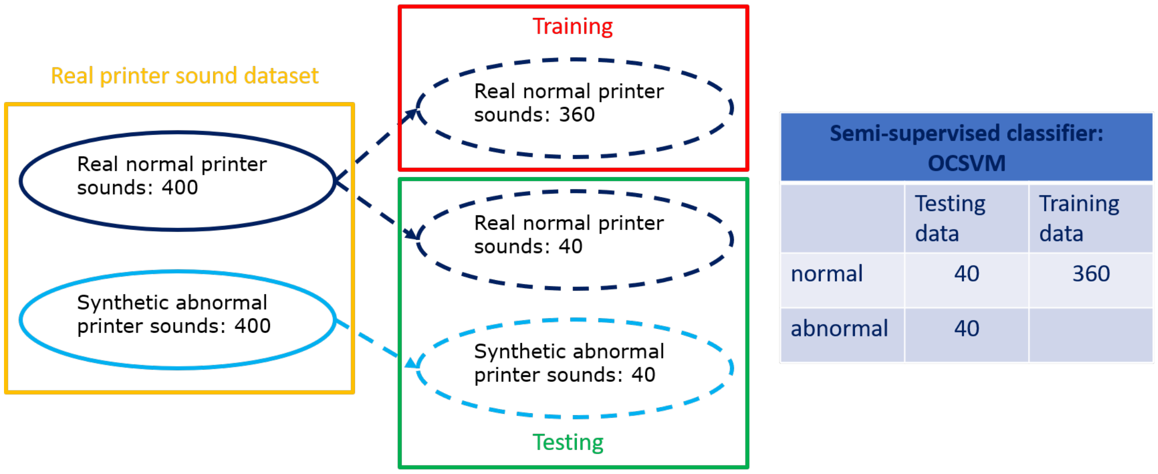

Real printer sound dataset
Real normal printer sounds: 400
Synthetic abnormal printer sounds: 400
Real normal printer sounds: 360
Real normal printer sounds: 40
Synthetic abnormal printer sounds: 40

Fig. 5.1.: Data arrangement part 1 for semi-supervised classifier.

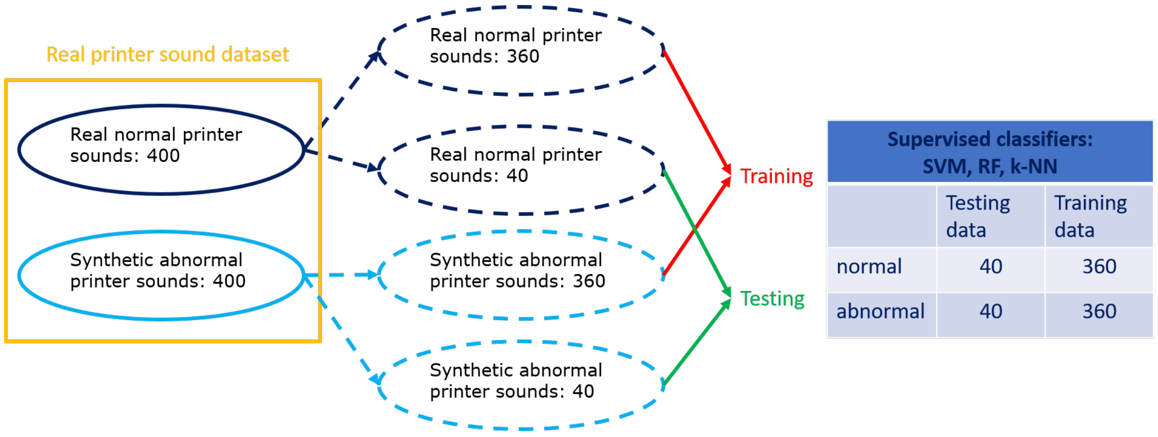
For supervised classifiers, SVM, RF, and k-NN, we randomly choose 360 within the 400 synthetic abnormal printer sounds as the abnormal training data and randomly choose


Training
Real printer sound dataset
Real normal printer sounds: 360
Real normal printer sounds: 400
Real normal printer sounds: 40
Synthetic abnormal printer sounds: 400
Synthetic abnormal printer sounds: 40
Testing

Semi-supervised classifier: OCSVM		
	Testing data	Training data
normal	40	360
abnormal	40	

Fig. 5.2.: Data arrangement part 2 for semi-supervised classifier.

360 within the 400 real normal printer sounds as the normal training data. The remaining 40 real normal and 40 synthetic abnormal printer sounds are the testing data as shown in Figure 5.3. Note that for supervised classifiers, they need both normal and abnormal


Real printer sound dataset
Real normal printer sounds: 360
Real normal printer sounds: 400
Real normal printer sounds: 40
Synthetic abnormal printer sounds: 400
Synthetic abnormal printer sounds: 360
Synthetic abnormal printer sounds: 40
Training
Testing

Supervised classifiers: SVM, RF, k-NN		
	Testing data	Training data
normal	40	360
abnormal	40	360

Fig. 5.3.: Data arrangement for supervised classifiers.

training data. On the other hand, semi-supervised classifier only needs normal training data. However, the testing data for all classifiers are the same. Here, all of the evaluation results are based on 10-fold cross validation as shown in Figure 5.4.

The data arrangement for the augmented dataset with semi-supervised classifier and supervised classifiers are shown in Figures 5.5 and 5.6, respectively. The data arrangement

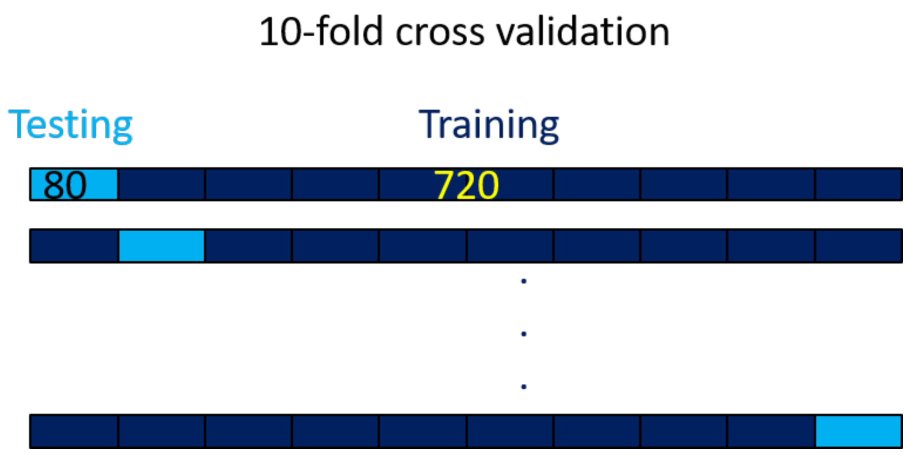


Fig. 5.4.: 10-fold cross validation

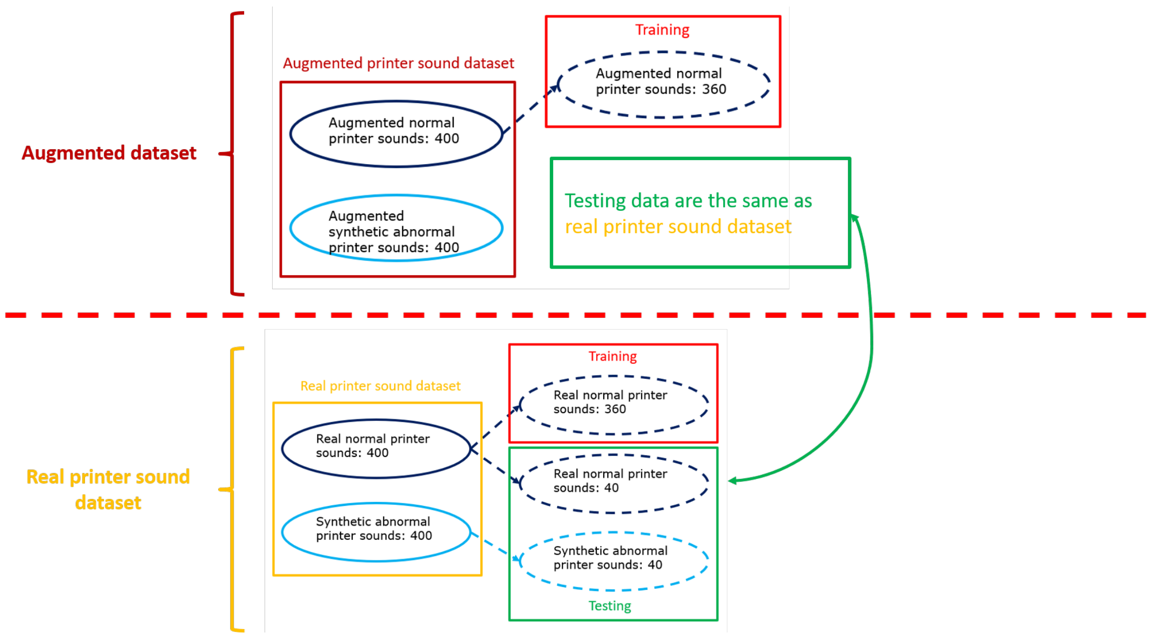
for the real printer sound dataset is below the red dash line and the data arrangement for the augmented dataset is above the red dash line. For semi-supervised classifier, OCSVM, the training data would be 360 augmented normal data and the testing data are the same as the real printer sound dataset as shown in Figure 5.2. For supervised classifiers, SVM, RF, and k-NN, the training data are 360 augmented normal data and 360 augmented abnormal. Note that the testing data are the same as the real printer sound dataset as shown in Figure 5.3.

5.2 Evaluation Methods

We use F-measure based on precision and recall with different weights to evaluate classifiers' performance. The value of F-measure is the higher the better. We will also show the accuracy performance at the same time. Table 5.1 shows the confusion matrix that we are using. Because we are dealing with anomaly detection task, the anomaly is more important

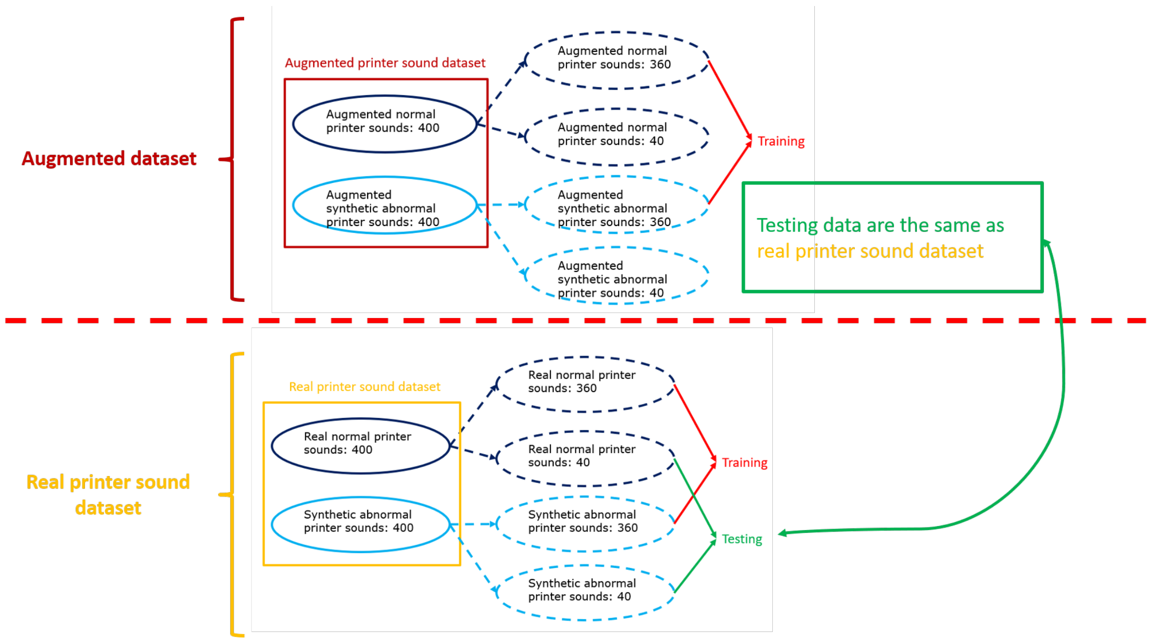
Confusion Matrix		Data predication	
		Normal	Abnormal
Data ground truth	Normal	TN	FP
	Abnormal	FN	TP

Table 5.1.: Confusion Matrix



Augmented dataset
Augmented printer sound dataset
Augmented normal printer sounds: 400
Augmented synthetic abnormal printer sounds: 400
Training
Augmented normal printer sounds: 360
Testing data are the same as real printer sound dataset
Real printer sound dataset
Real printer sound dataset
Real normal printer sounds: 400
Synthetic abnormal printer sounds: 400
Training
Real normal printer sounds: 360
Real normal printer sounds: 40
Synthetic abnormal printer sounds: 40
Testing

Fig. 5.5.: Data arrangement for the augmented dataset in semi-supervised classifier.



Augmented dataset
Augmented printer sound dataset
Augmented normal printer sounds: 400
Augmented synthetic abnormal printer sounds: 400
Augmented normal printer sounds: 360
Augmented normal printer sounds: 40
Augmented synthetic abnormal printer sounds: 360
Augmented synthetic abnormal printer sounds: 40
Training
Testing data are the same as real printer sound dataset
Real printer sound dataset
Real printer sound dataset
Real normal printer sounds: 400
Synthetic abnormal printer sounds: 400
Real normal printer sounds: 360
Real normal printer sounds: 40
Synthetic abnormal printer sounds: 360
Synthetic abnormal printer sounds: 40
Training
Testing

Fig. 5.6.: Data arrangement for the augmented dataset in supervised classifiers.

than normal. As a result, we take abnormal class as the positive case. Before look into the classifier performances, we demonstrate the definition of all evaluation methods. First of all, precision (P) is the fraction of predicted abnormal sample is abnormal ground truth as shown in Equation 5.1.

$$\text{Precision (P)} = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{5.1}$$

In other words, how confident can customers trust us? Next, recall is the fraction of abnormal ground truth is predicted as abnormal as shown in Equation 5.2.

$$\text{Recall (R)} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{5.2}$$

In other words, how confident can we trust the anomaly detection model? Based on Equations 5.1 and 5.2, we can further calculate the value of F-measure. F-measure [78, 79] provides a way to consider precision and recall at the same time with the different weight as shown in Equation 5.3,

$$\text{F-measure} = (1+\beta^2)\cdot\frac{\text{P}\cdot\text{R}}{\beta^2\cdot\text{P}+\text{R}} \tag{5.3}$$

where β is chosen such that recall is considered β times as important as precision. If β is less than 1, then precision is more important then recall. Vice versa, if β is larger than 1, then recall is more important than precision. In this dissertation, we use $\beta = 0.25$ because higher precision shows more persuasive evidence to our customers. Last evaluation method is accuracy as shown in Equation 5.4.

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{FN}+\text{TN}+\text{FP}} \tag{5.4}$$

Based on the description in Section 5.1, we do all experiments with balanced dataset, which means the number of the training data for both normal and abnormal class are the same. Furthermore, the number of the testing data are also the same for both classes. As a result, accuracy can also be considered as one of the evaluation method.

5.3 Results

Based on our system pipeline as shown in Figure 3.1, it shows two comparisons: Fixed feature with different inputs and different classifiers, and fixed input with different features and different classifiers.

5.3.1 Different Inputs vs Different Classifiers

In the first comparison, the three fixed features are: Detector with PCA, detector with mean and standard deviation, and MFCCs.

Evaluation Based on Detector with PCA

There are 660 different combinations of eleven training datasets and six classifiers based on detector with PCA as shown in Figure5.7. For the real printer sound dataset, it has real

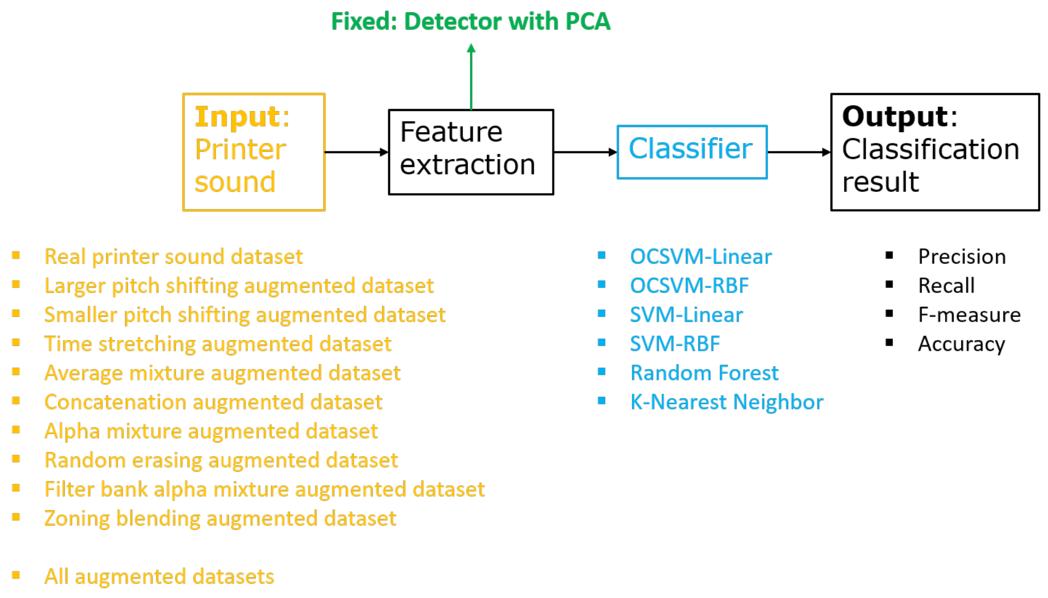

Fixed: Detector with PCA
Input: Printer sound
Feature extraction
Classifier
Output: Classification result
Real printer sound dataset
Larger pitch shifting augmented dataset
Smaller pitch shifting augmented dataset
Time stretching augmented dataset
Average mixture augmented dataset
Concatenation augmented dataset
Alpha mixture augmented dataset
Random erasing augmented dataset
Filter bank alpha mixture augmented dataset
Zoning blending augmented dataset
All augmented datasets
OCSVM-Linear
OCSVM-RBF
SVM-Linear
SVM-RBF
Random Forest
K-Nearest Neighbor
Precision
Recall
F-measure
Accuracy

Fig. 5.7.: Different inputs vs Different classifiers based on detector with PCA.

normal and synthetic abnormal audio samples. As the explanation in Section 5.1, we train the classifiers with real printer sounds and test them with real printer sounds. For the following nine augmented datasets, we train the classifiers with augmented printer sounds and test them with real printer sounds. For the last one, all augmented dataset, we train the classifiers with these nine augmented datasets and test them with real printer sounds. In other words, the first ten datasets train the classifiers with 360 normal and 360 abnormal, and test them with 40 normal and 40 abnormal. The last dataset, all augmented datasets, trains the classifiers with 3240 normal and 3240 abnormal, and test them with 40 normal and 40 abnormal. The evaluation results of F-measure, precision, recall, and accuracy are in Tables 5.2, 5.3, 5.4, and 5.5.

Based on the precision performance in Table 5.3, average mixture augmented dataset have the best performance with OCSVM with linear kernel. Filter bank alpha mixture augmented dataset has the best performance with OCSVM with RBF kernel, and random erasing augmented dataset has the closest performance compared with the best one within 1%. Real printer sound dataset has the best performance with SVM with linear kernel, and filter bank alpha mixture augmented dataset has the closest performance compared with the best one within 1%. Filter bank alpha mixture augmented dataset has the best performance with SVM with RBF kernel, and all augmented datasets have the closest performance compared with the best one within 1%. All augmented datasets have the best performance with RF, and alpha mixture augmented dataset has the closest performance compared with the best one within 1%. All augmented datasets have the best performance with k-NN, and real printer, alpha mixture, random erasing, and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 1%.

Based on the recall performance in Table 5.4, larger pitch shifting, time stretching, concatenation, alpha mixture, and random erasing augmented datasets have perfect recall with OCSVM with linear kernel. Concatenation augmented dataset has the best perofor-mance with OCSVM with RBF kernel. Real printer sound dataset has the best performance with SVM with linear kernel, and time stretching and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 1%. All augmented datasets have the best performance with SVM with RBF kernel. Random erasing augmented dataset has the best performance with RF, and larger pitch shifting and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 1%. All augmented datasets have the best performance with k-NN.

Based on the F-measure performance in Table 5.2,average mixture augmented dataset has the best performance with OCSVM with linear kernel. Random erasing augmented dataset has the best performance with OCSVM with RBF kernel. Real printer sound dataset has the best performance with SVM with linear kernel, and filter bank alpha mixture augmented dataset has the closest performance compared with the best one within 1%. Filter bank alpha mixture augmented dataset has the best performance with SVM with RBF kernel, and all augmented datasets have the closest performance compared with the best one within 1%. All augmented datasets have the best performance with RF, and alpha mixture, random erasing, and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 1%. All augmented datasets have the best performance with k-NN, and real printer sound dataset, random erasing, and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 1%.

Table 5.2 shows obviously that different combinations of augmentation method and classifier can improve the model performance based on the detector with PCA. More obviously, when we increase the number of the training data, the performance is better than the fewer one.

Evaluation Based on MFCCs

There are 660 different combinations of eleven training datasets and six classifiers based on MFCCs as shown in Figure 5.8. The data arrangement for training and testing data are the same in Section 5.3.1 The evaluation results of F-measure, precision, recall, and accuracy are in Tables 5.6, 5.7, 5.8, and 5.9.

Based on the precision performance in Table 5.7, average mixture augmented dataset has the best performance with OCSVM with linear kernel. All augmented datasets have the best performance with OCSVM with RBF kernel. Average mixture and concatena-

F-measure	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	67.88	82.2	96.49	92.69	97.07	99.31
Larger pitch shifting	62.01	86.46	91.57	92.16	91.38	93.28
Smaller pitch shifting	69.49	82.57	92.45	92.8	97.8	97.46
Time stretching	64.12	84.44	90.7	91.99	94.25	96.14
Average mixture	79.49	80.42	9373	92.31	96.82	93.91
Concatenation	56.94	84.87	91.23	91.52	90.58	92.39
Alpha mixture	61.61	86.33	92.48	92.18	98.34	98.39
Random erasing	67.97	88.06	92.62	92.32	98.02	98.76
Filter bank alpha mixture	66.4	85.8	96.47	98.24	97.81	98.57
Zoning blending	62.52	86.98	92.27	92.27	91.99	93.89
All augmented datasets	52.83	55.14	91.99	98.13	98.48	99.51

Table 5.2.: Comparison of classification F-measures ($\beta = 0.25$) achieved with different combinations of augmentation method and classifier based on the detector with PCA.

Precision	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	66.59 (3.58)	84.83 (4.24)	97.43 (2.32)	93.73 (2.53)	98.04 (3.52)	99.74 (0.77)
Larger pitch shifting	60.58 (2.18)	86.43 (3.73)	92.34 (3.34)	93.14 (2.87)	91.72 (3.14)	93.38 (2.75)
Smaller pitch shifting	68.2 (2.61)	84.51 (4.73)	93.19 (3.39)	93.06 (2.81)	98.93 (1.57)	97.81 (1.91)
Time stretching	62.73 (3.65)	83.95 (4.56)	91.24 (3.36)	92.94 (3.26)	94.85 (3.87)	96.33 (2.7)
Average mixture	78.69 (5.18)	86.14 (5.87)	94.81 (3.33)	93.37 (2.36)	98.65 (2.89)	94.99 (3.3)
Concatenation	55.45 (1.27)	84.32 (4.7)	92.02 (4.1)	92.29 (4.1)	91.05 (4.52)	93.17 (4.38)
Alpha mixture	60.17 (1.94)	87.07 (4.61)	93.35 (3.8)	93.13 (2.82)	99.58 (1.11)	98.95 (1.64)
Random erasing	66.63 (4.49)	88.52 (3.51)	93.36 (3.3)	93.37 (2.35)	98.79 (2.16)	99.11 (1.57)
Filter bank alpha mixture	65.06 (3.23)	88.91 (4.26)	97.41 (2.51)	99.63 (6.33)	98.61 (1.93)	98.84 (2.61)
Zoning blending	61.15 (3.25)	87.08 (2.99)	93.29 (2.94)	93.34 (2.37)	92.19 (4.75)	93.93 (2.61)
All augmented datasets	52.23 (1.59)	70.09 (10.95)	92.67 (3.67)	99.11 (1.92)	100 (0)	99.76 (0.73)

Table 5.3.: Comparison of classification precisions achieved with different combinations of augmentation method and classifier based on the detector with PCA. The value inside the parentheses is standard deviation within 10-fold cross validation.

Recall	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	98.75 (1.25)	55 (0)	83.75 (6.64)	78.75 (8)	85.8 (6.53)	93 (3.5)
Larger pitch shifting	100 (0)	87 (1.5)	80.9 (6.53)	79 (7.92)	86.23 (6.18)	91.75 (4.23)
Smaller pitch shifting	99.75 (0.75)	60.5 (4.3)	82.15 (6.67)	78.93 (8.21)	82.7 (6.88)	92.25 (4.49)
Time stretching	100 (0)	93.25 (1.15)	82.95 (6.8)	79.08 (7.92)	856.6 (6.31)	93.28 (3.93)
Average mixture	95 (0)	39 (3.74)	79.28 (7.52)	78.15 (8.13)	74.63 (7.23)	79.43 (6.88)
Concatenation	100 (0)	94.75 (0.75)	80.22 (7.93)	80.72 (8.29)	83.72 (7.4)	81.42 (7.3)
Alpha mixture	100 (0)	76 (3.74)	80.4 (6.97)	79.2 (7.94)	82 (6.75)	90.1 (4.31)
Random erasing	100 (0)	81.25 (8)	82.27 (5.25)	78.3 (8)	87.15 (6.46)	93.55 (3.81)
Filter bank alpha mixture	99 (1.22)	55 (0)	83.67 (6.66)	80.35 (6.33)	86.6 (6.03)	94.57 (3.48)
Zoning blending	97.5 (0)	85.5 (1.87)	78.6 (7.47)	77.95 (8.18)	82.87 (7.33)	93.2 (4.22)
All augmented datasets	64.75 (4.67)	12.5 (4.47)	82.5 (6.02)	84.75 (5.96)	82.5 (6.89)	95.75 (3.72)

Table 5.4.: Comparison of classification recalls achieved with different combinations of augmentation method and classifier based on the detector with PCA. The value inside the parentheses is standard deviation within 10-fold cross validation.

Accuracy (%)	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	74.39 (4)	72.5 (1.64)	90.75 (3.41)	86.75 (4.23)	92 (3.8)	96.38 (1.72)
Larger pitch shifting	67.35 (3.09)	86.58 (2.28)	87.11 (4.14)	86.63 (4.54)	89.23 (3.85)	92.63 (3.11)
Smaller pitch shifting	76.51 (2.79)	74.59 (2.65)	88.09 (4.17)	86.54 (4.49)	90.9 (3.56)	95.08 (2.39)
Time stretching	70.01 (4.68)	87.54 (3.19)	87.5 (4.21)	86.57 (4.64)	90.46 (4.09)	94.58 (2.86)
Average mixture	84.38 (4.03)	66.26 (2.27)	87.46 (4.26)	86.33 (4.34)	86.78 (3.84)	87.6 (3.88)
Concatenation	59.78 (2.8)	88.38 (3.17)	86.61 (4.67)	86.97 (4.87)	87.73 (4.87)	87.7 (4.61)
Alpha mixture	66.82 (2.63)	82.25 (2.97)	87.33 (4.29)	86.71 (4.46)	90.82 (3.39)	94.57 (2.45)
Random erasing	74.63 (4.91)	85.31 (4.05)	88.22 (3.64)	86.4 (4.34)	93.03 (3.53)	96.33 (1.96)
Filter bank alpha mixture	72.73 (3.83)	74 (1.47)	90.73 (3.88)	90.02 (3.24)	92.68 (3.28)	96.73 (2.32)
Zoning blending	67.55 (4.32)	86.35 (2.01)	86.48 (4.18)	86.22 (4.39)	87.88 (4.76)	93.6 (3.11)
All augmented datasets	52.75 (2)	53.63 (2.27)	88 (4.12)	92 (3.36)	91.25 (3.45)	97.75 (1.75)

Table 5.5.: Comparison of classification acciracies achieved with different combinations of augmentation method and classifier based on the detector with PCA. The value inside the parentheses is standard deviation within 10-fold cross validation.

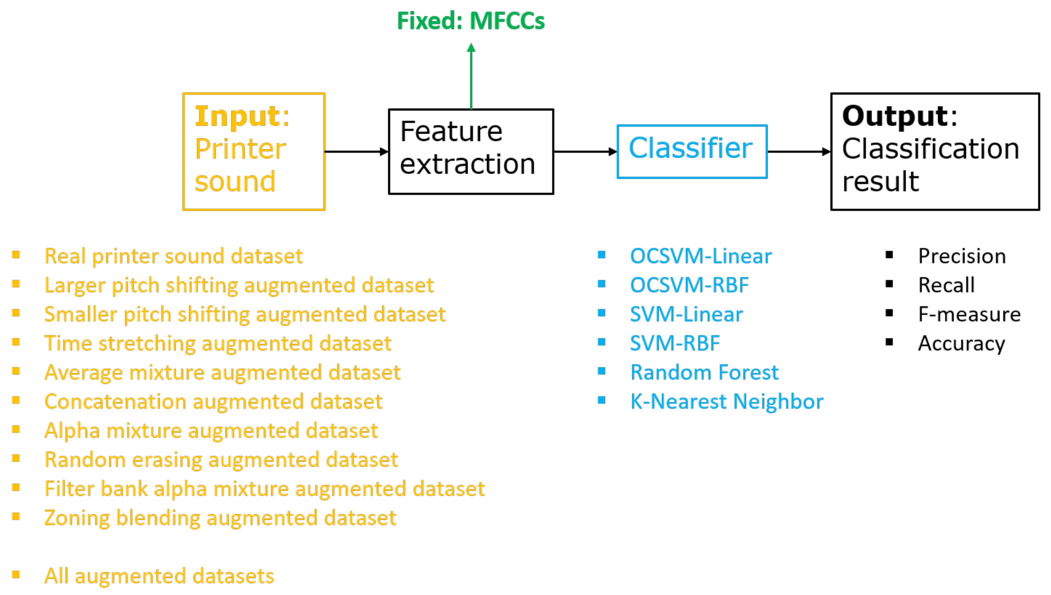

Fixed: MFCCs
Input: Printer sound
Feature extraction
Classifier
Output: Classification result
Real printer sound dataset
Larger pitch shifting augmented dataset
Smaller pitch shifting augmented dataset
Time stretching augmented dataset
Average mixture augmented dataset
Concatenation augmented dataset
Alpha mixture augmented dataset
Random erasing augmented dataset
Filter bank alpha mixture augmented dataset
Zoning blending augmented dataset
All augmented datasets
OCSVM-Linear
OCSVM-RBF
SVM-Linear
SVM-RBF
Random Forest
K-Nearest Neighbor
Precision
Recall
F-measure
Accuracy

Fig. 5.8.: Different inputs vs Different classifiers based on MFCCs

tion augmented datasets have the best performance with SVM with linear kernel, and time stretching, random erasing, and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 0.1%. All augmented datasets have the best performance with SVM with RBF kernel, and random erasing augmented dataset has the closest performance compared with the best one within 0.1%. All augmented datasets have the best performance with RF. All of the listed datasets have the perfect precision with k-NN.

Based on the recall performance in Table 5.8, except time stretching, average mixture, alpha mixture, and all augmented datasets, the remaining listed datasets have perfect recall with OCSVM with linear kernel. Except all augmented datasets, the remaining listed datasets have perfect recall with OCSVM with RBF kernel. Real printer sound dataset has the best performance with SVM with linear kernel, and random erasing and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 0.1%. All of the listed datasets have perfect recall with SVM with RBF kernel. Real printer sound dataset has the best performance with RF. Real printer sound dataset and all augmented datasets have the best performance with k-NN, and random erasing augmented dataset has the closest performance compared with the best one within 0.1%.

Based on the F-measure in Table 5.6, average mixture augmented dataset has the best performance with OCSVM with linear kernel. Real printer sound dataset has the best performance with OCSVM with RBF kernel. Filter bank alpha mixture augmented dataset has the best performance with SVM with linear kernel, and concatenation and random erasing augmented datasets have the closest performance compared with the best one within 0.1 %. All augmented datasets have the best performance with SVM with RBF kernel, and random erasing augmented datasets has the closest performance compared with the best one within 0.1%. All augmented datasets have the best performance with RF, and alpha mixture and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 0.1%. Real printer sound dataset, random erasing, and all augmented datasets have perfect F-measure with k-NN, and smaller pitch shifting, time stretching, concatenation, alpha mixture, and filter bank alpha mixture augmented datasets have the closest performance compared with the best one within 1%.

Table 5.6 shows obviously that different combinations of augmentation method and classifier can improve the model performance based on MFCCs. More obviously, when we increase the number of the training data, the performance is better than the fewer one

Evaluation Based on detector with mean and standard deviation

There are 660 different combinations of eleven training datasets and six classifiers based on detector with mean and standard deviation as shown in Figure 5.9. The data arrangement for training and testing data are the same as the illustration in Section 5.3.1. The evaluation results of F-measure, precision, recall, and accuracy are in Tables 5.10, 5.11, 5.12, and 5.13.

Based on the precision performance in Table 5.11, all augmented datasets has the best performance with OCSVM with linear kernel. All augmented datasets has the best performance with SVM with both linear and RBF kernels, and zoning blending augmented

F-measure	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	68.22	87.5	99.55	99.54	98.77	100
Larger pitch shifting	51.57	51.51	99.59	51.85	79.13	99.71
Smaller pitch shifting	52.28	51.51	99.51	66.1	97.91	99.93
Time stretching	84.19	74.35	98.42	99.54	98.68	99.98
Average mixture	87.18	53.69	97.93	84.98	98.93	99.59
Concatenation	70.32	62.09	99.83	96.65	97.48	99.97
Alpha mixture	82.6	74.18	99.67	99.3	99.27	99.98
Random erasing	72.45	77.21	99.95	99.68	99.15	100
Filter bank alpha mixture	73.11	66.42	99.9	99.27	99.26	99.97
Zoning blending	51.58	51.51	50.1	73.85	92.08	94.88
All augmented datasets	82.34	86.29	99.43	99.77	99.3	100

Table 5.6.: Comparison of classification F-measures ($\beta = 0.25$) achieved with different combinations of augmentation method and classifier based on MFCCs.

Precision	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	66.89 (3.71)	86.82 (4.69)	99.52 (1.43)	99.51 (0.98)	98.9 (1.55)	100 (0)
Larger pitch shifting	50.05 (0.18)	50 (0)	99.84 (0.6)	50.34 (0.5)	78.63 (14.05)	100 (0)
Smaller pitch shifting	50.76 (0.63)	50 (0)	99.6 (0.97)	64.73 (3.05)	99.32 (2.2)	100 (0)
Time stretching	83.84 (4.82)	73.17 (2.94)	99.96 (0.34)	99.51 (0.97)	99.48 (1.25)	100 (0)
Average mixture	87.01 (5.13)	52.18 (1.45)	100 (0)	84.19 (4.23)	99.76 (0.75)	100 (0)
Concatenation	69.04 (4.51)	60.64 (2.43)	100 (0)	96.44 (1.96)	97.77 (2.81)	100 (0)
Alpha mixture	81.81 (3.41)	73 (3.22)	99.89 (0.52)	99.25 (1.53)	99.71 (0.81)	100 (0)
Random erasing	71.21 (5.11)	76.12 (3.55)	99.95 (0.34)	99.65 (0.84)	99.38 (1.27)	100 (0)
Filter bank alpha mixture	71.9 (4.57)	65.06 (3.15)	99.9 (0.47)	99.23 (1.47)	99.62 (0.99)	100 (0)
Zoning blending	50.06 (0.18)	50 (0)	50.18 (5.89)	72.66 (4.84)	98.01 (3.87)	100 (0)
All augmented datasets	82.67 (5.43)	88.2 (8.91)	99.51 (0.98)	99.75 (0.73)	100 (0)	100 (0)

Table 5.7.: Comparison of classification precisions achieved with different combinations of augmentation method and classifier based on MFCCs. The value inside the parentheses is standard deviation within 10-fold cross validation.

Recall	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	100 (0)	100 (0)	100 (0)	100 (0)	96.78 (3.23)	100 (0)
Larger pitch shifting	100 (0)	100 (0)	95.67 (3.88)	100 (0)	88.1 (8.45)	95.37 (4.96)
Smaller pitch shifting	100 (0)	100 (0)	98 (1.76)	100 (0)	79.72 (12.49)	98.75 (1.75)
Time stretching	90 (0)	100 (0)	78.87 (8.06)	100 (0)	87.35 (8.7)	99.7 (0.81)
Average mixture	90 (0)	100 (0)	73.55 (8.7)	100 (0)	87.25 (8.21)	93.4 (4.27)
Concatenation	100 (0)	100 (0)	97.25 (1.92)	100 (0)	93 (6.1)	99.47 (1.02)
Alpha mixture	97.5 (0)	100 (0)	96.15 (3.28)	100 (0)	92.6 (5.61)	99.65 (0.86)
Random erasing	100 (0)	100 (0)	99.95 (0.35)	100 (0)	95.55 (3.97)	99.95 (0.35)
Filter bank alpha mixture	100 (0)	100 (0)	99.9 (0.48)	100 (0)	93.77 (5.04)	99.55 (0.96)
Zoning blending	100 (0)	100 (0)	48.77 (6.6)	100 (0)	46.8 (19.92)	52.17 (22.37)
All augmented datasets	77.25 (11.64)	64 (15.37)	98.25 (1.95)	100 (0)	89.25 (7.66)	100 (0)

Table 5.8.: Comparison of classification recalls achieved with different combinations of augmentation method and classifier based on MFCCs. The value inside the parentheses is standard deviation within 10-fold cross validation.

Accuracy (%)	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	75.01 (4.36)	92.24 (3.26)	99.75 (0.75)	99.75 (0.5)	97.84 (1.8)	100 (0)
Larger pitch shifting	50.11 (0.35)	50 (0)	97.96 (1.96)	50.67 (0.97)	79.91 (11.37)	97.68 (2.48)
Smaller pitch shifting	51.5 (1.22)	50 (0)	98.8 (0.98)	72.58 (3.64)	89.56 (6.22)	99.37 (0.87)
Time stretching	86.15 (3.15)	81.56 (2.7)	89.42 (4.04)	99.75 (0.5)	93.45 (4.45)	99.85 (0.4)
Average mixture	88.1 (3.1)	54.11 (2.65)	86.77 (4.35)	90.46 (2.95)	93.51 (4.04)	96.7 (2.13)
Concatenation	77.25 (5.09)	67.42 (3.35)	98.62 (0.96)	98.13 (1.08)	95.4 (3.27)	99.73 (0.51)
Alpha mixture	87.81 (2.52)	81.37 (2.99)	98.02 (1.72)	99.61 (0.8)	96.16 (2.8)	99.82 (0.43)
Random erasing	79.41 (5.35)	84.17 (3.09)	99.95 (0.24)	99.82 (0.43)	97.47 (2.19)	99.97 (0.17)
Filter bank alpha mixture	80.16 (4.68)	72.97 (3.65)	99.9 (0.33)	99.6 (0.76)	96.71 (2.63)	99.77 (0.48)
Zoning blending	50.12 (0.37)	50 (0)	50.05 (5.61)	80.88 (4.54)	72.88 (9.85)	76.08 (11.18)
All augmented datasets	80.5 (6.45)	78.12 (9.25)	98.87 (1.03)	99.87 (0.37)	94.62 (3.83)	100 (0)

Table 5.9.: Comparison of classification accuracies achieved with different combinations of augmentation method and classifier based on MFCCs. The value inside the parentheses is standard deviation within 10-fold cross validation.

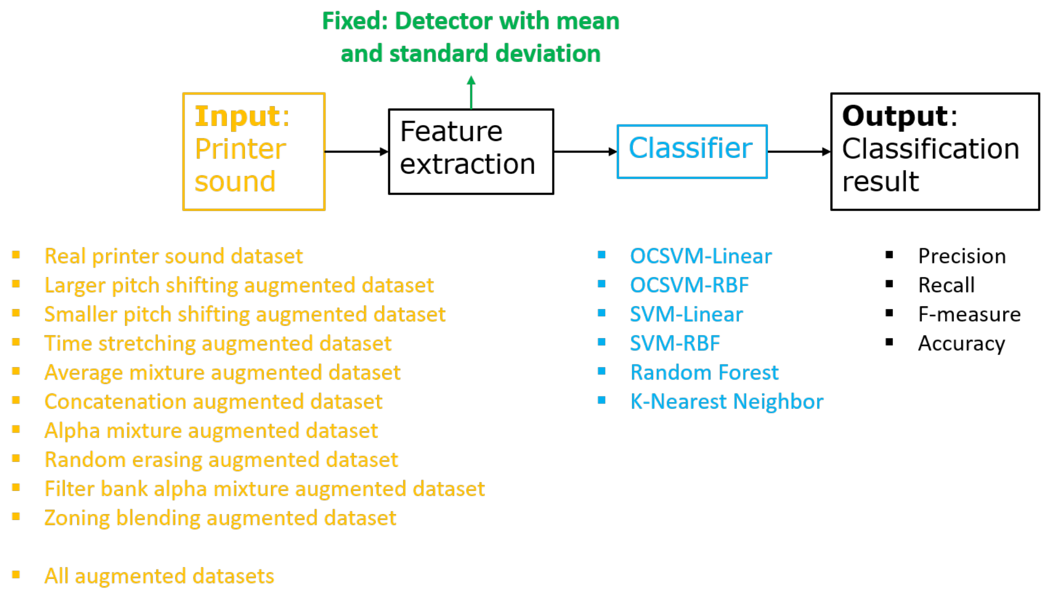

Fixed: Detector with mean and standard deviation
Input: Printer sound
Feature extraction
Classifier
Output: Classification result
Real printer sound dataset
Larger pitch shifting augmented dataset
Smaller pitch shifting augmented dataset
Time stretching augmented dataset
Average mixture augmented dataset
Concatenation augmented dataset
Alpha mixture augmented dataset
Random erasing augmented dataset
Filter bank alpha mixture augmented dataset
Zoning blending augmented dataset
All augmented datasets
OCSVM-Linear
OCSVM-RBF
SVM-Linear
SVM-RBF
Random Forest
K-Nearest Neighbor
Precision
Recall
F-measure
Accuracy

Fig. 5.9.: Different inputs vs Different classifiers based on detector with mean and standard deviation.

dataset has the closest performance compared with the best one within 1%. Filter bank alpha mixture augmented dataset has the best performance with RF, and random erasing, zoning blending, and all augmented datasets have the closest performance compared with the best one within 1%. All augmented datasets have the best performance with k-NN. Note that OCSVM with RBF kernel didn't work for the detector with mean and standard deviation feature.

Based on the recall performance in Table 5.12, average mixture, concatenation, and alpha mixture augmented datasets have perfect recall with OCSVM with linear kernel. Average mixture augmented dataset has the best performance with SVM with both linear and RBF kernels. Random erasing augmented dataset has the best performance with RF, and real printer sound dataset has the closest performance compared with the best one within 1%. All augmented datasets has the best performance with k-NN, and filter bank alpha mixture augmented dataset has the closest performance compared with the best one within 1%. Note that OCSVM with RBF kernel didn't work for the detector with mean and standard deviation feature.

Based on the F-measure in Table 5.10, all augmented datasets have the best performance with OCSVM with linear kernel. All augmented datasets have the best performance with SVM with both linear and RBF kernels, and zoning blending augmented dataset has the closest performance compared with the best one within 1 %. Filter bank alpha mixture augmented dataset has the best performance with RF, and random erasing augmented dataset and all augmented datasets have the closest performance compared with the best one within 1%. All augmented datasets have the best performance with k-NN.

Table 5.10 shows obviously that different combinations of augmentation method and classifier can improve the model performance based on the detector with mean and standard deviation. More obviously, when we increase the number of the training data, the performance is better than the fewer one.

5.3.2 Different Features vs Different Classifiers

There are 18 different combinations of three features and six classifiers based on the input with real printer sound dataset as shown in Figure 5.10. The evaluation results of

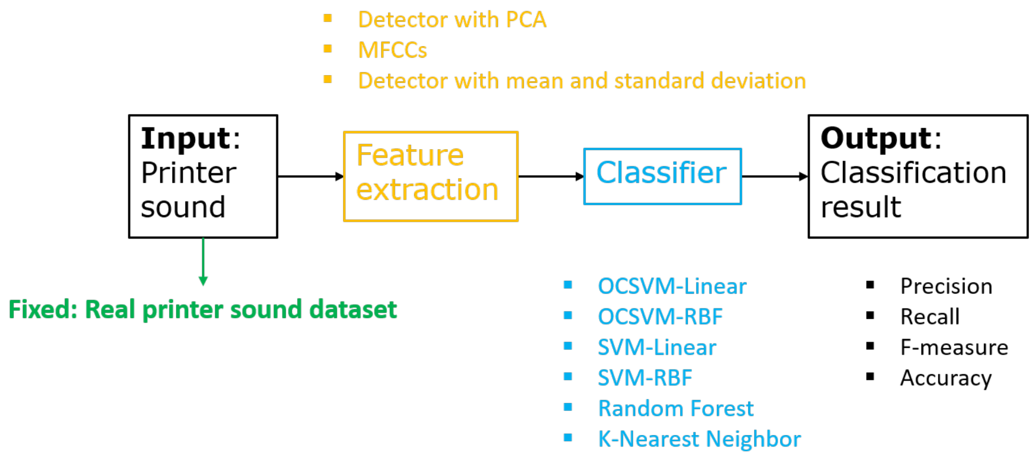


Fig. 5.10.: Different feature vs Different classifier based on real printer sound dataset.

F-measure, precision, recall, and accuracy are in Tables 5.14, 5.15, 5.16, and 5.17.

F-measure	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	66.12	NAN	87.96	81.54	92.62	95.12
Larger pitch shifting	60.72	NAN	85.1	74.68	91.33	90.74
Smaller pitch shifting	60.23	NAN	87.62	85.89	92.64	93.81
Time stretching	62.68	14.03	87.62	84.1	91.41	90.61
Average mixture	56.59	10.24	60.81	57.31	76.85	89.71
Concatenation	60.13	9.16	NAN	NAN	15.74	64.52
Alpha mixture	61.74	NAN	83.05	63.26	89.53	92.85
Random erasing	65.8	1.53	88.39	81.26	93.22	94.08
Filter bank alpha mixture	66.45	NAN	86.82	83.29	93.47	95.63
Zoning blending	62.45	NAN	89.2	87.52	92.44	91.14
All augmented datasets	69.29	1.11	89.81	88.15	93.04	97.6

Table 5.10.: Comparison of classification F-measures ($\beta = 0.25$) achieved with different combinations of augmentation method and classifier based on the detector with mean and standard deviation.

Precision	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	64.98 (4.62)	0	87.52 (4.93)	80.81 (5.19)	92.36 (3.91)	95 (2.97)
Larger pitch shifting	59.45 (2.59)	0	84.61 (4.55)	73.53 (3.48)	91.12 (4.27)	90.43 (4.33)
Smaller pitch shifting	58.8 (2.34)	0	87.28 (4.1)	85.49 (4.58)	92.48 (4.25)	93.78 (3.83)
Time stretching	61.41 (2.15)	19.71 (7.04)	87.41 (5.1)	83.69 (5.43)	91.2 (4.45)	90.36 (4.01)
Average mixture	55.1 (1.85)	12.7 (2.8)	59.37 (2.64)	55.88 (1.83)	79.66 (14.07)	90.81 (4.12)
Concatenation	58.67 (2.53)	11.8 (6.61)	0	0	21 (31.13)	76.59 (14.21)
Alpha mixture	60.29 (2.28)	0	82.36 (3.86)	61.98 (3.62)	89.7 (3.89)	92.85 (3.21)
Random erasing	64.57 (4.16)	2.26 (7.63)	87.94 (4.23)	80.43 (5.14)	92.96 (4.08)	93.89 (3.36)
Filter bank alpha mixture	65.27 (4.26)	0	86.31 (4.2)	82.79 (5.57)	93.28 (3.51)	95.5 (3.05)
Zoning blending	61.23 (3.38)	0	89.16 (3.77)	87.19 (4.27)	92.29 (4.46)	90.98 (4.12)
All augmented datasets	69.41 (5.61)	1.42 (4.28)	89.68 (3.14)	87.83 (4.34)	92.88 (3.76)	97.54 (1.52)

Table 5.11.: Comparison of classification precisions achieved with different combinations of augmentation method and classifier based on the detector with mean and standard deviation. The value inside the parentheses is standard deviation within 10-fold cross validation.

Recall	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	91.75 (1.14)	0	95.75 (2.96)	95.25 (2.83)	97 (2.64)	97.25 (2.07)
Larger pitch shifting	92.5 (0)	0	93.7 (2.56)	99.6 (1.1)	94.82 (4.14)	95.87 (2.38)
Smaller pitch shifting	98.75 (1.25)	0	93.45 (3.6)	92.87 (3.01)	95.3 (3.67)	94.27 (2.67)
Time stretching	93.25 (1.14)	2.5 (4.43)	91.25 (3.61)	91.1 (4.54)	94.98 (3.89)	94.85 (2.93)
Average mixture	100 (0)	2.5 (0)	99.25 (1.14)	99.75 (0.75)	49.12 (14.5)	75.12 (6.05)
Concatenation	100 (0)	2 (1)	0	0	3.15 (5.6)	18.32 (7.01)
Alpha mixture	100 (0)	0	95.92 (3.05)	94.25 (2.56)	86.87 (6.27)	92.8 (4.91)
Random erasing	94.75 (0.75)	0.25 (0.75)	96.25 (3.11)	97.3 (1.72)	97.63 (2.45)	97.2 (2.29)
Filter bank alpha mixture	93.25 (1.14)	0	96.05 (2.96)	92.23 (4.07)	96.5 (3.27)	97.78 (1.9)
Zoning blending	91.5 (1.22)	0	89.8 (4.33)	93.25 (4.05)	95.03 (3.99)	93.58 (4.11)
All augmented datasets	67.5 (5.59)	0.25 (0.75)	92 (3.84)	93.5 (3)	95.75 (3.54)	98.5 (2)

Table 5.12.: Comparison of classification recalls achieved with different combinations of augmentation method and classifier based on the detector with mean and standard deviation. The value inside the parentheses is standard deviation within 10-fold cross validation.

Accuracy (%)	Classifiers					
Training dataset	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Real printer	70.83 (4.53)	44.96 (1.35)	90.87 (3.26)	86.12 (4.27)	94.38 (2.37)	96 (1.45)
Larger pitch shifting	64.56 (3.28)	47.56 (1.02)	88.13 (285)	81.72 (3.01)	92.63 (2.27)	92.76 (2.94)
Smaller pitch shifting	64.66 (3.29)	45.31 (1.47)	89.78 (2.69)	88.41 (3.22)	93.63 (2.38)	93.93 (2.57)
Time stretching	67.25 (2.61)	45.48 (1.95)	88.86 (3.08)	86.4 (3.52)	92.73 (2.31)	92.26 (2.61)
Average mixture	59.15 (3.1)	42.18 (2.16)	65.48 (3.86)	60.4 (2.95)	69.05 (7.68)	83.66 (3.13)
Concatenation	64.62 (3.79)	43.52 (2.13)	44.88 (2.45)	50 (0)	48.27 (3.58)	56.66 (4.16)
Alpha mixture	66.96 (3.12)	47.61 (1.44)	87.57 (3.03)	67.95 (4.48)	88.38 (3.82)	92.76 (2.7)
Random erasing	71.1 (4.4)	45.05 (2.2)	91.42 (3.17)	86.56 (3.72)	95 (2.37)	95.35 (1.66)
Filter bank alpha mixture	71.53 (4.3)	46.33 (1.42)	90.28 (2.97)	86.28 (4.19)	94.7 (2.35)	96.53 (1.98)
Zoning blending	66.57 (3.87)	47.57 (1.02)	89.82 (1.55)	89.62 (2.68)	93.38 (2.55)	92.02 (2.63)
All augmented datasets	68.75 (4.71)	45.12 (1.17)	90.62 (2.03)	90.12 (2.52)	94.12 (2.68)	98 (1.27)

Table 5.13.: Comparison of classification acciracies achieved with different combinations of augmentation method and classifier based on the detector with mean and standard deviation. The value inside the parentheses is standard deviation within 10-fold cross validation.

Based on the precision performance in Table 5.15, MFCCs feature has the best performance with all six classifiers. Based on the recall performance in Table 5.16, except RF, MFCCs feature has the best performance the remaining listed classifiers. Detector with mean and standard deviation feature has the best performance with RF. Based on the F-measure performance in Table 5.14, MFCCs feature has the best performance with all six classifiers.

Table 5.14 shows obviously that different combinations of feature and classifier can improve the model performance. Note that although the detector with mean and standard deviation has lower F-measure value, the computation cost of it is the lowest one within these three features. Here shows the trade-off between the computation cost and the model performance.

5.4 Conclusion

We have developed an anomaly detection system to diagnose the printer health. Our proposed anomaly detection pipeline consists of three parts: First, data preparation for synthetic abnormal sounds and the augmented sounds; Second, feature extraction based on detector with principal component analysis, Mel frequency cepstral coefficients, and detector with mean and standard deviation; Third, use one of six different classifiers that we explored; Fourth, categorize the input printer sound into normal or abnormal class. The proposed augmented dataset can improve the performance of our anomaly detection model. For the semi-supervised OCSVM with linear kernel, the average mixture augmented dataset with detector with PCA and MFCCs has the largest improvement over the real printer sound dataset. For the four fully supervised classifiers, the random erasing and filter bank alpha mixture augmentation methods work well with all features, except for the detector with mean and standard deviation. Finally, we investigated novel augmentation methods, different features, and different classifiers to improve the model performance, and find the most appropriate one.

F-measure	Classifiers					
Features	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Detector with PCA	67.88	82.2	96.49	92.69	97.07	99.31
MFCCs	68.22	87.5	99.55	99.54	98.77	100
Detector with mean and standard deviation	66.12	NAN	87.96	81.54	92.62	95.12

Table 5.14.: Comparison of classification F-measures ($\beta = 0.25$) achieved with different combinations of feature and classifier based on real printer sound dataset.

Precision	Classifiers					
Features	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Detector with PCA	66.59 (3.58)	84.83 (4.24)	97.43 (2.32)	93.73 (2.53)	98.04 (3.52)	99.74 (0.77)
MFCCs	66.89 (3.71)	86.82 (4.69)	99.52 (1.43)	99.51 (0.98)	98.9 (1.55)	100(0)
Detector with mean and standard deviation	64.98 (4.62)	0	87.52 (4.93)	80.81 (5.19)	92.36 (3.91)	95 (2.97)

Table 5.15.: Comparison of classification precisions achieved with different combinations of feature and classifier based on real printer sound dataset. The value inside the parentheses is standard deviation within 10-fold cross validation.

Recall	Classifiers					
Features	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Detector with PCA	98.75 (1.25)	55 (0)	83.75 (6.64)	78.75 (8)	85.8 (6.53)	93 (3.5)
MFCCs	100 (0)	100 (0)	100 (0)	100 (0)	96.78 (3.23)	100 (0)
Detector with mean and standard deviation	91.75 (1.14)	0	95.75 (2.96)	95.25 (2.83)	97 (2.64)	97.25 (2.07)

Table 5.16.: Comparison of classification recalls achieved with different combinations of feature and classifier. The value inside the parentheses is standard deviation within 10-fold cross validation.

Accuracy	Classifiers					
Features	OCSVM-Linear	OCSVM-RBF	SVM-Linear	SVM-RBF	RF	k-NN(k=5)
Detector with PCA	74.39 (4)	72.5 (1.64)	90.75 (3.41)	86.75 (4.23)	92 (3.8)	96.38 (1.72)
MFCCs	75.01 (4.36)	92.23 (3.25)	99.75 (0.75)	99.75 (0.75)	97.83 (1.8)	100 (0)
Detector with mean and standard deviation	70.83 (4.53)	44.96 (1.35)	90.87 (3.26)	86.12 (4.27)	94.38 (2.37)	96 (1.45)

Table 5.17.: Comparison of classification accuracies achieved with different combinations of feature and classifier based on real printer sound dataset. The value inside the parentheses is standard deviation within 10-fold cross validation.

6. CONCLUSION AND CONTRIBUTION

We summarize the contributions of this dissertation as follows:

- We have developed a novel, aperiodic, dispersed-dot halftoning solution for a simulated multilevel CMY printer with 16 levels per channel.
- Our proposed imaging pipeline consists of:
  - First, transform input sRGB to YyCxCz linearized uniform color space.
  - Next, we do the gamut mapping to fit the source gamut into the printer gamut also in the YyCxCz color space.
  - Finally, we do vector error diffusion that maps modified continuous-tone image values to the nearest output color on a $16 \times 16 \times 16$ grid.
- It yields good quality output images but the improvement over FSED provided by TDFED is not as significant as it is for a binary output device.
- We have expanded an anomaly detection system to diagnose the printer health.
- Our proposed anomaly detection pipeline consists three parts:
  - Data preparation for synthetic abnormal sounds and the augmented sounds.
    - Explored or developed eight different augmentation methods
  - Feature extraction based on
    - Detector with principal component analysis
    - Mel frequency cepstral coefficients
    - Detector with mean and standard deviation
  - Explored the accuracy of six different classifiers

  - Third, categorize the input printer sound into normal or abnormal class.

- The proposed augmented dataset can improve the performance of our anomaly detection model:

  - For the semi-supervised OCSVM with linear kernel, the average mixture augmented dataset with detector with PCA and MFCCs has the largest improvement over the real printer sound dataset.
  - For the four fully supervised classifiers, the random erasing and filter bank alpha mixture augmentation methods work well with all features, except for the detector with mean and standard deviation.

- Finally, we investigated novel augmentation methods, different features, and different classifiers to improve the model performance, to find the most appropriate one.

REFERENCES

[1] T. J. Flohr, B. W. Kolpatzik, R. Balasubramanian, D. A. Carrara, C. A. Bouman, and J. P. Allebach, “Model-based color image quantization,” *Journal of Electronic Imaging*, vol. 1913, pp. 270–282, 1993.

[2] R. W. Floyd and L. Steinberg, “Adaptive algorithm for spatial greyscale,” *Proceedings of the Society of Information Display*, vol. 17, no. 2, pp. 75–77, 1976.

[3] P. Li and J. P. Allebach, “Tone-dependent error diffusion,” *IEEE Transactions on Image Processing*, vol. 13, no. 2, pp. 201–215, 2004.

[4] V. Monga, N. Damera-Venkata, and B. L. Evans, “Design of tone-dependent color-error diffusion halftoning systems,” *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 198–211, 2006.

[5] G. Sarailidis and I. Katsavounidis, “A multiscale error diffusion technique for digital multitoning,” *IEEE Transactions on Image Processing*, vol. 21, no. 5, pp. 2693–2705, 2012.

[6] J.-M. Guo, J.-Y. Chang, Y.-F. Liu, G.-H. Lai, and J.-D. Lee, “Tone-replacement error diffusion for multitoning,” *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 4312–4321, 2015.

[7] H. Yu, K. Inoue, K. Hara, and K. Urahama, “Color error diffusion based on neugebauer model,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 99, no. 9, pp. 1758–1761, 2016.

[8] M. Wolski, J. P. Allebach, and C. A. Bouman, “Gamut mapping squeezing the most out of your color system,” *Color and Imaging Conference*, vol. 1994, no. 1, pp. 89–92, 1994.

[9] R. S. Gentile, J. P. Allebach, and E. Walowit, “A comparison of techniques for color gamut mismatch compensation,” *Journal of Electronic Imaging*, vol. 1077, pp. 342–355, 1989.

[10] G. Sharma and R. Bala, *Digital Color Imaging Handbook*. CRC press, 2002.

[11] T. Fawcett and F. Provost, “Adaptive fraud detection,” *Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 291–316, 1997.

[12] R. Arroyo, J. J. Yebes, L. M. Bergasa, I. G. Daza, and J. Almazán, “Expert video-surveillance system for real-time detection of suspicious behaviors in shopping malls,” *Expert Systems with Applications*, vol. 42, no. 21, pp. 7991–8005, 2015.

[13] T. Hayashi, T. Komatsu, R. Kondo, T. Toda, and K. Takeda, “Anomalous sound event detection based on wavenet,” *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 2494–2498, 2018.

[14] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, 2012.

[15] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," *IEEE international conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4277–4280, 2012.

[16] M. Espi, M. Fujimoto, K. Kinoshita, and T. Nakatani, "Exploiting spectro-temporal locality in deep learning based acoustic event detection," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2015, no. 1, p. 26, 2015.

[17] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *The Detection and Classification of Acoustic Scenes and Events (DCASE)*, pp. 1–5, 2017.

[18] N. Sawhney and P. Maes, "Situational awareness from environmental sounds," *Project Rep. for Pattie Maes*, pp. 1–7, 1997.

[19] D. W. Scott, "Outlier detection and clustering by partial mixture modeling," *COMPSTAT 2004—Proceedings in Computational Statistics*, pp. 453–464, 2004.

[20] Y. Kawaguchi, R. Tanabe, T. Endo, K. Ichige, and K. Hamada, "Anomaly detection based on an ensemble of dereverberation and anomalous sound extraction," *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 865–869, 2019.

[21] H. Phan, L. Hertel, M. Maass, R. Mazur, and A. Mertins, "Representing nonspeech audio signals through speech classification models," *16th Annual Conference of the International Speech Communication Association (ISCA)*, pp. 3441–3445, 2015.

[22] W. Choi, S. Park, D. K. Han, and H. Ko, "Acoustic event recognition using dominant spectral basis vectors," *16th Annual Conference of the International Speech Communication Association (ISCA)*, pp. 2002–2006, 2015.

[23] J. Beltrán, E. Chávez, and J. Favela, "Scalable identification of mixed environmental sounds, recorded from heterogeneous sources," *Pattern Recognition Letters*, vol. 68, pp. 153–160, 2015.

[24] T. N. Sainath, A.-r. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for lvcsr," *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8614–8618, 2013.

[25] S. Chu, S. Narayanan, and C.-C. J. Kuo, "Environmental sound recognition with time–frequency audio features," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 6, pp. 1142–1158, 2009.

[26] S. Pancoast and M. Akbacak, "Bag-of-audio-words approach for multimedia event classification," *13th Annual Conference of the International Speech Communication Association (ISCA)*, pp. 2105–2108, 2012.

[27] H. Lim, M. J. Kim, and H. Kim, "Robust sound event classification using lbp-hog based bag-of-audio-words feature representation," *16th Annual Conference of the International Speech Communication Association (ICSA)*, pp. 3325–3329, 2015.

[28] X. Lu, P. Shen, Y. Tsao, C. Hori, and H. Kawai, "Sparse representation with temporal max-smoothing for acoustic event detection," *16th Annual Conference of the International Speech Communication Association (ICSA)*, pp. 1176–1180, 2015.

[29] Y. Xue, N. Kim, X. Wang, J. P. Allebach, J. S. Bolton, P. Davies, G. Chiu, K. Ferguson, D. Klaisle, and M. Shaw, "Digital signal processing for laser printer noise source detection and identification," in *the Proceedings of Noise-Con*, 2019.

[30] X. Wang, "Harmonic scrubber for detected modulation frequencies." Masters Thesis, Purdue University, West Lafayette, IN 47907, May, 2019.

[31] P. Welch, "The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms," *IEEE Transactions on Audio and Electroacoustics*, vol. 15, no. 2, pp. 70–73, 1967.

[32] J. G. Proakis, M. Salehi, N. Zhou, and X. Li, *Communication Systems Engineering*. Prentice Hall New Jersey, 1994.

[33] S. Raschka and V. Mirjalili, *Python Machine Learning*. Packt Publishing, 2017.

[34] I. Jolliffe, *Principal Component Analysis*. Springer, 2011.

[35] T. Zhang and B. Yang, "Big data dimension reduction using pca," *IEEE International Conference on Smart Cloud*, pp. 152–157, 2016.

[36] C. Yuan, X. Sun, and R. Lv, "Fingerprint liveness detection based on multi-scale lpq and pca," *China Communications*, vol. 13, no. 7, pp. 60–65, 2016.

[37] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[38] K. K. Vasan and B. Surendiran, "Dimensionality reduction using principal component analysis for network intrusion detection," *Perspectives in Science*, vol. 8, pp. 510–512, 2016.

[39] D. Ye, Y. Fuh, Y. Zhang, G. Hong, and K. Zhu, "Defects recognition in selective laser melting with acoustic signals by SVM based on feature reduction," *IOP Conference Series: Materials Science and Engineering*, vol. 436, no. 1, p. 012020, 2018.

[40] Z.-J. Chuang and C.-H. Wu, "Emotion recognition using acoustic features and textual content," *IEEE International Conference on Multimedia and Expo (ICME)*, vol. 1, pp. 53–56, 2004.

[41] S. Abidin, R. Togneri, and F. Sohel, "Acoustic scene classification using joint time-frequency image-based feature representations," *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6, 2018.

[42] X. Huang, A. Acero, H.-W. Hon, and R. Foreword By-Reddy, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall New Jersey, 2001.

[43] C. Ittichaichareon, S. Suksri, and T. Yingthawornsuk, "Speech recognition using MFCC," *International Conference on Computer Graphics, Simulation and Modeling (ICGSM'2012)*, pp. 28–29, 2012.

[44] F. S. Al-Anzi and D. AbuZeina, “The capacity of mel frequency cepstral coefficients for speech recognition,” *International Journal of Computer and Information Engineering*, vol. 11, no. 10, pp. 1149–1153, 2017.

[45] L. E. Boucheron, P. L. De Leon, and S. Sandoval, “Low bit-rate speech coding through quantization of mel-frequency cepstral coefficients,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 610–619, 2011.

[46] S. Aziz, M. Awais, T. Akram, U. Khan, M. Alhussein, and K. Aurangzeb, “Automatic scene recognition through acoustic classification for behavioral robotics,” *Electronics*, vol. 8, no. 5, p. 483, 2019.

[47] Y. Chen, Q. Guo, X. Liang, J. Wang, and Y. Qian, “Environmental sound classification with dilated convolutions,” *Applied Acoustics*, vol. 148, pp. 123–132, 2019.

[48] D. L. Moore, Z. Hamilton, and R. J. Haddad, “Pico-grid smart home energy management system using mel frequency cepstral coefficients,” *IEEE SoutheastCon*, pp. 1–6, 2019.

[49] Q. Mei, M. Gül, and M. Boay, “Indirect health monitoring of bridges using mel-frequency cepstral coefficients and principal component analysis,” *Mechanical Systems and Signal Processing*, vol. 119, pp. 523–546, 2019.

[50] W.-Y. Chuang, Y.-L. Tsai, and L.-H. Wang, “Leak detection in water distribution pipes based on cnn with Mel frequency cepstral coefficients,” *Proceedings of International Conference on Innovation in Artificial Intelligence*, pp. 83–86, 2019.

[51] M. Usman, Z. Ahmad, and M. Wajid, “Dataset of raw and pre-processed speech signals, Mel frequency cepstral coefficients of speech and heart rate measurements,” *IEEE International Conference on Signal Processing, Computing and Control (ISPCC)*, pp. 376–379, 2019.

[52] B. P. Bogert, “The quefrency alanysis of time series for echoes; cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking,” *Time Series Analysis*, pp. 209–243, 1963.

[53] D. G. Childers, D. P. Skinner, and R. C. Kemerait, “The cepstrum: A guide to processing,” *Proceedings of the IEEE*, vol. 65, no. 10, pp. 1428–1443, 1977.

[54] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic press, 2014.

[55] N. Ahmed, T. Natarajan, and K. R. Rao, “Discrete cosine transform,” *IEEE Transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[56] L. Nanni, Y. M. Costa, D. R. Lucio, C. N. Silla Jr, and S. Brahnam, “Combining visual and acoustic features for audio classification tasks,” *Pattern Recognition Letters*, vol. 88, pp. 49–56, 2017.

[57] K. J. Piczak, “ESC: Dataset for environmental sound classification,” *Proceedings of International Conference on Multimedia*, pp. 1015–1018, 2015.

[58] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, “Estimating the support of a high-dimensional distribution,” *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[59] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999.

[60] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[61] I. Guyon, B. Boser, and V. Vapnik, "Automatic capacity tuning of very large VC-dimension classifiers," *Advances in Neural Information Processing Systems*, pp. 147–155, 1993.

[62] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[63] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[64] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, no. Feb, pp. 207–244, 2009.

[65] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, vol. 117, 2013.

[66] X. Cui, V. Goel, and B. Kingsbury, "Data augmentation for deep neural network acoustic modeling," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 9, pp. 1469–1477, 2015.

[67] N. Takahashi, M. Gygli, B. Pfister, and L. Van Gool, "Deep convolutional neural networks and data augmentation for acoustic event detection," *arXiv preprint arXiv:1604.07160*, 2016.

[68] H. Zeinali, L. Burget, and J. Cernocky, "Convolutional neural networks and x-vector embedding for dcase2018 acoustic scene classification challenge," *arXiv preprint arXiv:1810.04273*, 2018.

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

[70] K. Xu, D. Feng, H. Mi, B. Zhu, D. Wang, L. Zhang, H. Cai, and S. Liu, "Mixup-based acoustic scene classification using multi-channel convolutional neural network," *Pacific Rim Conference on Multimedia*, pp. 14–23, 2018.

[71] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," *arXiv preprint arXiv:1708.04896*, 2017.

[72] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[73] S. Gharib, H. Derrar, D. Niizumi, T. Senttula, J. Tommola, T. Heittola, T. Virtanen, and H. Huttunen, "Acoustic scene classification: A competition review," *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2018.

[74] S. Wei, K. Xu, D. Wang, F. Liao, H. Wang, and Q. Kong, "Sample mixed-based data augmentation for domestic audio tagging," *arXiv preprint arXiv:1808.03883*, 2018.

[75] H. Inoue, “Data augmentation by pairing samples for images classification,” *arXiv preprint arXiv:1801.02929*, 2018.

[76] S. Abidin, R. Togneri, and F. Sohel, “Enhanced LBP texture features from time frequency representations for acoustic scene classification,” *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 626–630, 2017.

[77] S. Abidin, X. Xia, R. Togneri, and F. Sohel, “Local binary pattern with random forest for acoustic scene classification,” *IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, 2018.

[78] N. Chinchor and B. Sundheim, “MUC-5 evaluation metrics,” *Proceedings of the Conference on Message Understanding*, pp. 69–78, 1993.

[79] Y. Sasaki, “The truth of the F-measure,” *Teach Tutor Master*, 2007.

VITA

Chin-Ning Chen received both her BS and MS in electrical and computer engineering from the National Chiao Tung University, Hsinchu, Taiwan, in 2014 and 2016, respectively. Currently, she is a Ph.D. student in Purdue University, West Lafayette, IN, where she has been since 2016. Her research interests include color management , halftoning and acoustic anomaly detection.