SPARSE DISCRETE WAVELET DECOMPOSITION
AND
FILTER BANK TECHNIQUES FOR SPEECH RECOGNITION

by

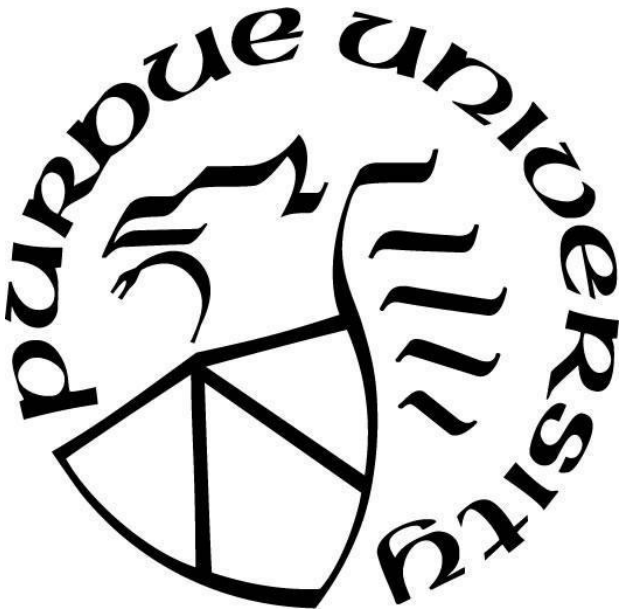
Jingzhao Dai

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science in Electrical and Computer Engineering



School of Electrical & Computer Engineering

Hammond, Indiana

May 2019

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Lizhe Tan, Chair

Department of Electrical and Computer Engineering

Dr. Bin Chen

Department of Electrical and Computer Engineering

Dr. Yao Xu

Department of Electrical and Computer Engineering

Approved by:

Dr. Vijay Devabhaktuni

Head of the department, Electrical and Computer Engineering

Dedicated to my parents Jinsong Dai and Yuanyuan Fang

ACKNOWLEDGMENTS

Here, I wish to thank my academic advisor, Dr. Li-zhe Tan for his patient guidance and scientific insight. Since I began my thesis work under his guidance, I have learned much knowledge and many methods for the research. I believe that it will be beneficial for my future career life. Through this thesis project, I not only learn engieneering expertise, but also the ethic, dedication and passion for research.

I would also like to thank my other committee members, Dr. Bin Chen and Dr. Yao Xu for their feedbacks. I also wish to thank all my group members in my research for their help and brainstorming.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

MFCC	Mel Frequency Cepstral Coefficients
SDWD	Sparse Discrete Wavelet Decomposition
BPFB	Bandpass Filter Banks
SVM	Support Vector Machine
SVC	Support Vector Classification
RF	Random Forest
KNN	K Nearest Neighbors
CNN	Convolutional Neural Network

.

ABSTRACT

Author: Dai, Jingzhao. MSECE
Institution: Purdue University
Degree Received: May 2019
Title: Sparse Discrete Wavelet Decomposition and Filter Bank Techniques for Speech Recognition
Committee Chair: Li-Zhe Tan

Speech recognition is widely applied to translation from speech to related text, voice driven commands, human machine interface and so on [1]-[8]. It has been increasingly proliferated to Human's lives in the modern age. To improve the accuracy of speech recognition, various algorithms such as artificial neural network, hidden Markov model and so on have been developed [1], [2].

In this thesis work, the tasks of speech recognition with various classifiers are investigated. The classifiers employed include the support vector machine (SVM), k-nearest neighbors (KNN), random forest (RF) and convolutional neural network (CNN). Two novel features extraction methods of sparse discrete wavelet decomposition (SDWD) and bandpass filtering (BPF) based on the Mel filter banks [9] are developed and proposed. In order to meet diversity of classification algorithms, one-dimensional (1D) and two-dimensional (2D) features are required to be obtained. The 1D features are the array of power coefficients in frequency bands, which are dedicated for training SVM, KNN, and RF classifiers while the 2D features are formed both in frequency domain and temporal variations. In fact, the 2D feature consists of the power values in decomposed bands versus consecutive speech frames. Most importantly, the 2D features with geometric transformation are adopted to train CNN.

Speech recordings including males and females are from the recorded data set as well as the standard data set. Firstly, the recordings with little noise and clear pronunciation are applied with the proposed feature extraction methods. After many trials and experiments using this dataset, a high recognition accuracy is achieved. Then, these feature extraction methods are further applied to the standard recordings having random characteristics with ambient nosie and unclear pronunciation. Many experiment results validate the effectiveness of the proposed feature extraction techniques.

CHAPTER 1. INTRODUCTION

1.1 Motivations

Due to an increasing need of speech recognition techniques, solving more corresponding issues or possible improvements become more and more important. To improve the accuracy and robust of speech recognition, choosing methods for feature extraction is a vital part of speech recognition. In speech feature extraction, the decomposition method and referred frequency bands are required to be considered.

For speech decomposition, the wavelet decomposition is a common method. However, this method has issues in which huge computation load is required and each decomposed bandwidth usually cannot match the referred frequency bands. And for the referred frequency bands, the Mel filter banks are widely applied. These banks are the series of perceptual frequency bands with human auditory information [9]. It is also a widely accepted feature extraction method to improve speech recognition performance [1], [3]. Based on the two shortages of wavelet decomposition referring to Mel filter banks, the sparse discrete wavelet decomposition (SDWD) and bandpass filter banks (BPFB) are proposed.

The choices of neural network algorithms and applied data sets are important. In this thesis work, data sets are speech recordings for ten digits ("zero" to "nine"). The speech recordings include two females and two males. Another applied data set is chosen from the standard data sets especially for speech research [10]. For network alogorithms, the several standard network algorithms such as support vector machine (SVM), random forest (RF), k nearest neighbors (KNN) and convolutional neural networks (CNN) are selected. To formulate features for CNN deep learning, the features are required to form in two dimensions; and for the other networks, the features are formed in one dimension.

1.2 Objectives

In this thesis work, the whole goal is to prove the outperformance of speech recognition using sparse discrete decomposition (SDWD) and bandpass filter banks (BPFB) with convolutional neural network (CNN) deep learning. To prove that, SDWD and BPFB features are also taken as

the inputs to support vector machine (SVM), random forest (RF), k neareat neigbors (KNN), and corresponding performances are compared to SDWD-CNN and BPFB-CNN. In order to realize the whole goal, many objectives are required to be realized. The first objective is to make decomposed frequency bands to match Mel filter banks as close as possible. The second one is to formulate the features for the CNN structure.

For the first objective, that is, matching the Mel filter bank specifications, many modifications have been done in adjusting the decomposed frequency bands. For BPFB, each edge value of decomposition is set to the same as Mel filter bank edges. In addition, each decomposed bandwidth is divided by a constant value to make the center frequency match the Mel band center frequency better; and for SDWD, to match each center and edge frequency values well, the Mel filter banks are modified to obtain the modified referred bands for SDWD. For the second objective, that is, formulating features for CNN structure, the main task is to transform the shape of each two-dimensional feature from rectangle to square. The employed method for shape transformation is bilinear interpolation.

Note that before all of objectives mentioned above, preprocessing to each speech signal is commonly required to be conducted. In this work, normalization and voice activity detection (VAD) are conducted to extract the voiced region in speech containing ambient noise. In addition, because of different sampling rates used in the two data sets, resampling is also necessary.

1.3 Organization of Thesis

The structure of this paper is organized as follows. Chapter 2 introduces the SDWD and BPFB algorithms and procedures for preprocessing. In Chapter 3, more details for all employed network algorithms are investigated and various performances in terms of various parameters are compared. Finally, Chapter 4 presents the conclusion of all testing accuracy for each of the data sets. In addition, some future works are listed at the end of Chapter 4.

CHAPTER 2. SPEECH RECOGNITION FEATURES

2.1 Mel Frequency Cepstral Coefficients (MFCC)

The Mel frequency cepstral coefficients (MFCC) are widely used for extracting speech features. MFCC are obtained from a warped spectrum for perceptual human auditory information and also includes short time Fourier transform (STFT). It is a consensus that employing perceptual human auditory information can efficiently improve the accuracy and robust of the speech recognition. The process of obtaining Mel filter bank frequency specifications is described below.

Mel filter bank frequency specifications are obtained from Mel scale, which is shown as Equation (2.1),

$$m = Mel(f) = 1125 \times \ln\left(1 + \frac{f}{700}\right), \tag{2.1}$$

where f is the frequency in Hz. When the Mel scale is given, the frequency can be derived by the Mel scale inverse transformation shown in (2.2),

$$f = Mel^{-1}(m) = 700 \times \exp\left[\left(\frac{m}{1125}\right) - 1\right]. \tag{2.2}$$

To describe the sensitivity of the human auditory system, the Mel scale is distributed equally. There are three steps for employing Mel filter banks. The first step is mapping the lowest and highest frequency values to the corresponding Mel scale values using (2.1) to determine the range of the Mel scale. Secondly, the edge values can be derived by uniformly distributing Mel scale domain [9]. The number of edge values obtained depends on how many filters proposed to decompose the speech signal. Thirdly, the frequency edge values can be obtained by remapping Mel scale values using (2.2). The obtained frequency values are exactly the Mel filter bank specifications. The specifications are required when sparse discrete wavelet decomposition and bandpass filter banks are employed.

In this thesis work, the sampling rate is set to 8000 Hz. Therefore, the lowest and highest frequency values are separately 300 Hz and 4000 Hz. There are 32 filters to decompose the signal. Thus, the Mel scale edge values are given by

$$
\begin{aligned}
m = [&401.3\ 454.0\ 506.8\ 559.5\ 612.3\ 665.0\ 717.8\ 770.6... \\
&823.3\ 876.1\ 928.8\ 981.6\ 1034.4\ 1087.1\ 1139.9... \\
&1192.6\ 1245.4\ 1298.1\ 1350.9\ 1403.7\ 1456.4\ 1509.2... \\
&1561.9\ 1614.7\ 1667.4\ 1720.2\ 1773.0\ 1825.7\ 1878.5... \\
&1931.2\ 1984.0\ 2036.8\ 2089.5\ 2142.3].
\end{aligned}
\tag{2.3}
$$

Using m, the frequency edge values remapped from the values in m are shown as

$$
\begin{aligned}
f = [&300\ 348.031\ 398.331\ 451.065\ 506.331\ 564.250\ 624.950... \\
&688.565\ 755.234\ 825.104\ 898.328\ 975.069\ 1055.493... \\
&1139.780\ 1228.113\ 1320.687\ 1417.706\ 1519.383... \\
&1625.942\ 1737.617\ 1854.654\ 1977.310\ 2105.855... \\
&2240.572\ 2381.757\ 2529.721\ 2684.789\ 2847.303... \\
&3017.619\ 3196.112\ 3383.176\ 3579.220\ 3784.678\ 4000].
\end{aligned}
\tag{2.4}
$$

Finally, the obtained Mel filter bank specifications in (2.4) are shown in the Figure 2.1, where all specification values are rounded to integers.

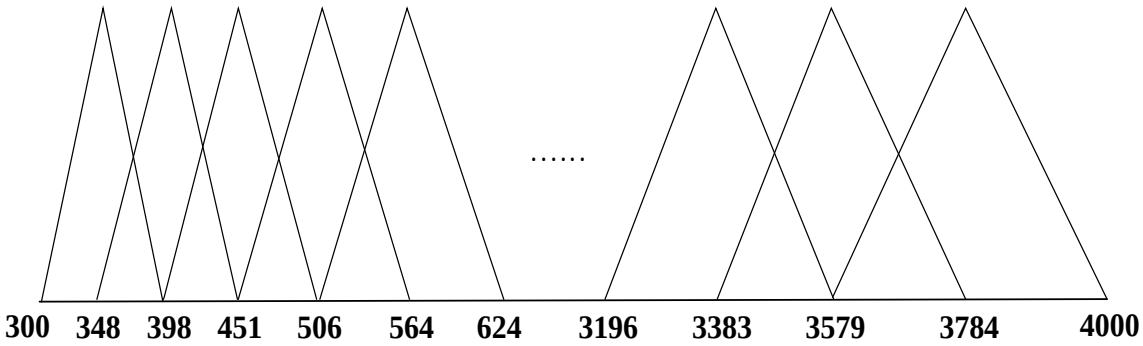


Figure 2.1 32 Mel filter banks.

Note that there exist overlaps among filters as shown in Figure 2.1 due to the filter roll-off nature. For instance, the band 30 with the range from 3383.176 Hz to 3784.678 Hz has an overlapped frequency part with the band 31 from 3579.220 Hz to 4000.000 Hz.

2.2 Speech Discrete Wavelet Transform

The discrete wavelet packet transform (DWPT) is also a method for retrieving band signals which are available for features extracting. However, the computation load of employing DWPT is huge. To solve this issue, the sparse discrete wavelet decomposition (SDWD) is developed. But

the size of each sub-band derived from SDWD is constant and is difficult to match the edge values of the Mel filter banks very well. To further improve speech recognition accuracy, the referred frequency bands for SDWD are modified based on the Mel filter banks. In that case, the overlaps among decompositions can also be improved.

2.2.1 Wavelet Decomposition

The discrete wavelet decomposition has been widely applied to decompose the speech signal into low and high frequency components. The wavelet decomposition with two levels is shown in Figure 2.2 [11].

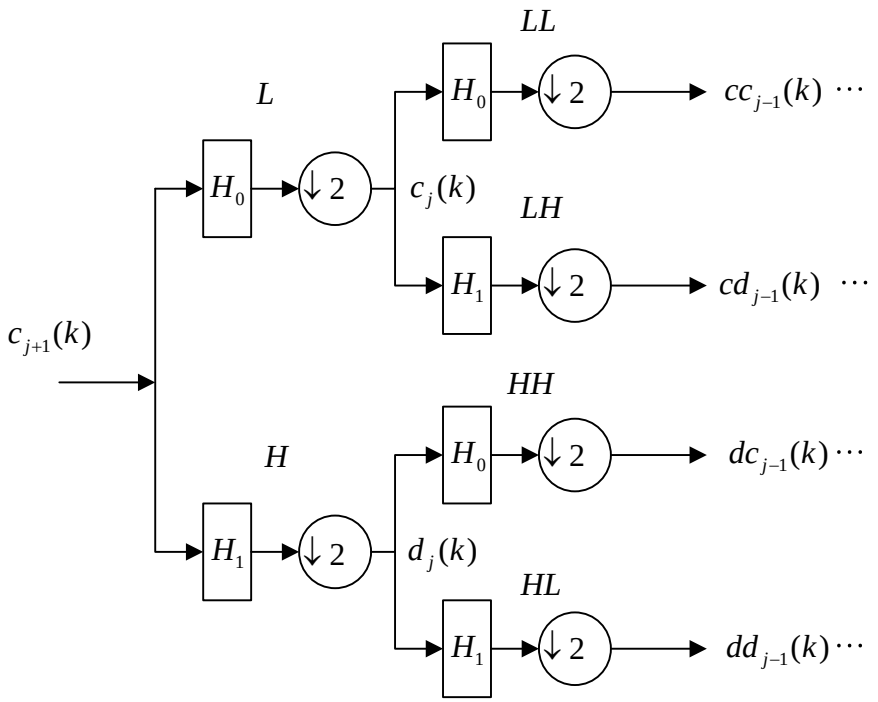


Figure 2.2 Discrete wavelet decomposition for two levels.

In Figure 2.2, the down-sampled low frequency component $c_j(k)$ and high-pass component $d_j(k)$ are decomposed from the signal $c_{j+1}(k)$ at the $(j+1)$ level [9].

$$c_j(k) = \sum_{m=-\infty}^{\infty} c_{j+1}(m) h_0(m-2k), \tag{2.5}$$

$$d_j(k) = \sum_{m=-\infty}^{\infty} c_{j+1}(m) h_1(m-2k), \tag{2.6}$$

where $h_0(k)$ and $h_1(k)$ are respectively low-pass and high-pass wavelet filters. The relation between them is given by

$$h_1(k) = (-1)^k h_0(N-1-k). \tag{2.7}$$

In the beginning of the discrete wavelet decomposition, the original speech signal $s(k)$ is set to

$$c_{j+1}(k) = s(k). \tag{2.8}$$

To the next level, the decomposed high-pass signal will be kept while the low-pass signal will be continually decomposed. As illustrated in the Figure 2.3 [11], for the speech samples with the decomposition for ten levels, it can be noticed that there are 11 band coefficients at level 10 for $\{c_0\}, \{d_0\}, \{d_1\}, \ldots, \{d_9\}$, which separately represent information for each band.

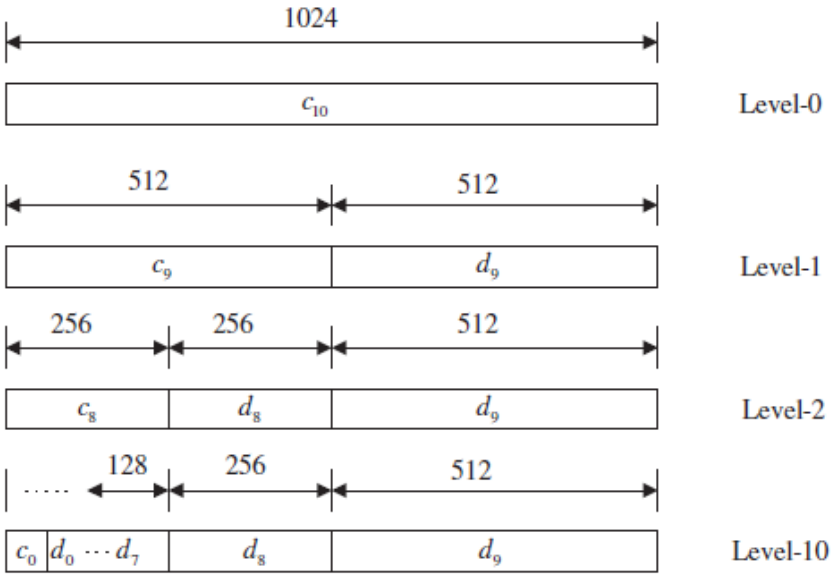


Figure 2.3 DWT coefficient layout for the ten-level decomposition.

In Figure 2.3, it can be observed that the size of each wavelet band is constant, while the size of the Mel filter banks in Figure 2.1 is not. Thus, the accuracy of the speech recognition using wavelet decomposition is limited because the wavelet bands can't match the Mel filter banks very well. In addition, the computation load of wavelet decomposition is huge. To tackle both issues,

the sparse discrete decomposition based on the modified Mel filter banks is proposed in this thesis work.

2.2.2 Sparse Discrete Wavelet Decomposition (SDWD)

Based on the first issue on dealing with huge computation, the sparse discrete wavelet decomposition (SDWD) is developed. The SDWD is applied with the dedicated decomposition path depending on the Gray code sequences [9]. Those decomposition can cover the modified Mel filter frequency bands as close as possible.

For the modified Mel filter banks, as shown in Table 2.1, the lower edge, center frequency and upper edge in each modified Mel filter bandwidth are given by

$$Mbw(i) = [(f(i-1) + f(i))/2,\ f(i),\ (f(i) + f(i+1))/2], \tag{2.9}$$

where $Mbw(i)$ represents the $i\text{-}th$ modified Mel filter bandwidth, $f(i)$ is the $i\text{-}th$ number in the array of f and represents the center frequency in $Mbw(i)$. Table 2.1 lists all modified bandwidths and corresponding paths. The number of symbols (L and H) in each path indicates the number of levels for signal decomposition. Through experiments, the more levels for decomposition, the better the decomposed bands can match $Mbw(i)$. However, the disadvantage is that more levels for decomposition may require a longer length of speech frame. Otherwise, there will be no speech samples in the decomposed band at the highest level.. Thus, the highest decomposition level is restricted to seven (7). In other words, the most number of symbols for paths in Table 2.1 [12] is seven (7).

Depending on $Mbw(i)$ in Table 2.1, the sparse discrete decomposition can search the corresponding decomposed bands via the paths following Gray code sequence. For example, the band 11, [936.6985, 975.0687, 1015.2810] in Table 2.1, can be approximately covered by a decomposition path of LLHLLL (937.5-1000 Hz); and the band 15, [1274.3998, 1320.6869, 1369.1963] is achieved by two decomposition paths: LHHHLH and LHHHLLH with the combined band from 1375 to 1468.75 Hz [12].

Table 2.1 Sparse Discrete Wavelet Decomposition

Band number i	$Mbw(i)$ & decomposition paths
1	[324.0064, 348.0128, 373.1719], LLLHHH
2	[373.1719, 398.3309, 424.6979], LLLHLH
3	[424.6979, 451.0649, 478.6978], LLLHLLH
4	[478.6978, 506.3308, 535.2905], LLLHLLL, LLHHLLL
5	[535.2905, 564.2501, 594.6004], LLHHLLH, LLHHLHH
6	[594.6003, 624.9504, 656.7577], LLHHLHL, LLHHHHL
7	[656.7577, 688.5650, 721.8995], LLHHHHH, LLHHHLH
8	[721.8995, 755.2339, 790.1689], LLHHHLL, LLHLHLL
9	[790.1689, 825.1038, 861.7161], LLHLHLH, LLHLHHH
10	[861.7161, 898.3284, 936.6895], LLHLHHL, LLHLLH
11	[936.6985, 975.0687, 1015.2810], LLHLLL
12	[1015.2810, 1055.4934, 1097.6365], LHHLLL, LHHLLHH
13	[1097.6365, 1139.7797, 1183.9462], LHHLLHL, LHHLHH
14	[1183.9462, 1228.1127, 1274.3998], LHHLHL, LHHHHLL
15	[1274.3998, 1320.6869, 1369.1963], LHHHHLH, LHHHHH
16	[1369.1963, 1417.7058, 1468.5443], LHHHLH, LHHHLLH
17	[1468.5443, 1519.3828, 1572.6623], LHHHLLL, LHLHLL
18	[1572.6623, 1625.9417, 1681.7792], LHLHLH, LHLHHH
19	[1681.7792, 1737.6167, 1796.1352], LHLHHL, LHLLHL
20	[1796.1352, 1854.6536, 1915.9817], LHLLHH, LHLLLHL
21	[1915.9817, 1977.3098, 2041.5824], LHLLLHH, LHLLLL, HHLLLLL
22	[2041.5824, 2105.8550, 2173.2136], HHLLLLH, HHLLLH, HHLLHH
23	[2173.2136, 2240.5721, 2311.1647], HHLLHL, HHLHHL
24	[2311.1647, 2381.7573, 2455.7393], HHLHHH, HHLHLH
25	[2455.7393, 2529.7212, 2607.2553], HHLHLL, HHHHLL, HHHHLHH
26	[2607.2553, 2684.7893, 2766.0460], HHHHLHL, HHHHH
27	[2766.0460, 2847.3026, 2932.4607], HHHLH, HHHLLH
28	[2932.4607, 3017.6187, 3106.8654], HHHLLL, HLHLL
29	[3106.8654, 3196.1121, 3289.6438], HLHLH, HLHHHLL
30	[3289.6438, 3383.1755, 3481.1979], HLHHHLH, HLHHHH, HLHHL
31	[3481.1979, 3579.2203, 3681.9491], HLLHL, HLLHHH
32	[3681.9491, 3784.6778, 3892.3389], HLLHHL, HLLLH

Note that the higher level the path belongs to, the more accurately the decomposition edge will match the edges of $Mbw(i)$. However, if a speech signal is decomposed for many levels, the information in the highest decomposition level can be lost. Thus, in this thesis work, only levels 1 to 7 are proposed.

2.3 Bandpass Filter Banks (BPFB)

As an alternative method to decompose the speech signal, band pass filtering is proposed. In this thesis work, the bandpass filter banks using bilinear transformation (BLT) [11] is employed. The BLT requires a process to pre-warp the frequency specifications of the analog filter. The prewarping equation is shown as

$$\omega_a = \frac{2}{T}\tan(\frac{\omega_d T}{2}), \tag{2.10}$$

where ω_d is the digital frequency, ω_a is the analog frequency and T is the sampling period. The BLT is defined as

$$s = \frac{2}{T}\frac{z-1}{z+1}, \tag{2.11}$$

where z and s are the complex variables in the z-plane and s-plane.

In this work, the upper and lower cut-off frequency values as well as the center frequency can be determined based on the Mel filter band edge values in (2.4). Then they are first warped to analog frequency values. The second-order analog filter is designed based on the first-order Butterworth lowpass prototype filter and prototype transformation, shown as the substitution (2.13) in (2.12).

$$H_p(s) = \frac{1}{s+1}, \tag{2.12}$$

$$s = \frac{s^2 + \omega_o{}^2}{sW}, \tag{2.13}$$

where W is the analog frequency bandwidth and ω_o is the analog center frequency. Thus, the secnd-order digital filter using BLT can be derived by substituting the BLT to the second-order analog bandpass filter, that is,

$$H(z) = \left.\frac{Ws}{s^2 + Ws + \omega_o{}^2}\right|_{s=\frac{2}{T}\frac{z-1}{z+1}}.$$

For implementation, thirty-two (32) IIR second-order bandpass filter banks (BPFB) are used for decomposing the speech signal. The decomposed signal is achieved as

$$x(k) = x_1(k) + x_2(k) + ... + x_{32}(k), \tag{2.14}$$

where $x_1(n)$, $x_2(n)$,…, and $x_{32}(n)$ are the decomposed signals respectively obtained from the corresponding bandpass filters, that is,

$$x_1(n) = h_1(n) * x(n), \tag{2.15}$$

$$x_2(n) = h_2(n) * x(n), \tag{2.16}$$

…

$$x_{32}(n) = h_{32}(n) * x(n), \tag{2.17}$$

where $h_1(n)$, $h_2(n)$, … and $h_{32}(n)$ are the impulse responses of band 1, band 2, …, and band 32 which are respectively labeled as “BPF1”, “BPF2” and “BPF32” in Figure 2.4.

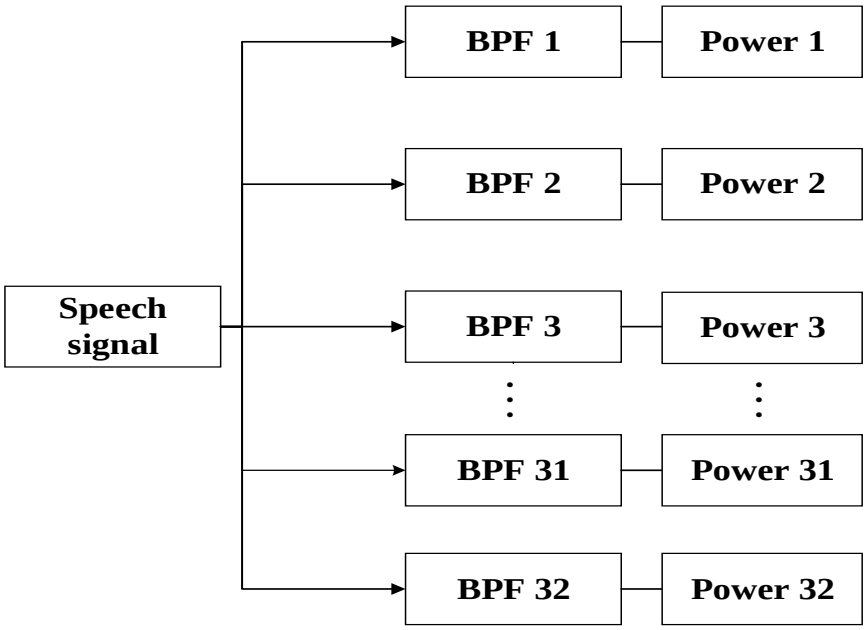


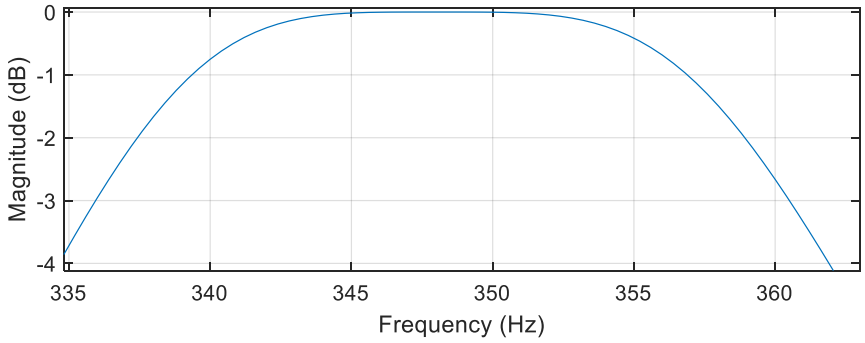
Figure 2.4 Speech decomposition using the bandpass filter banks.

In this work, the center frequency for band i is required to match the center frequency value $f(i)$ in (2.4); and the bandwidth is yielded as

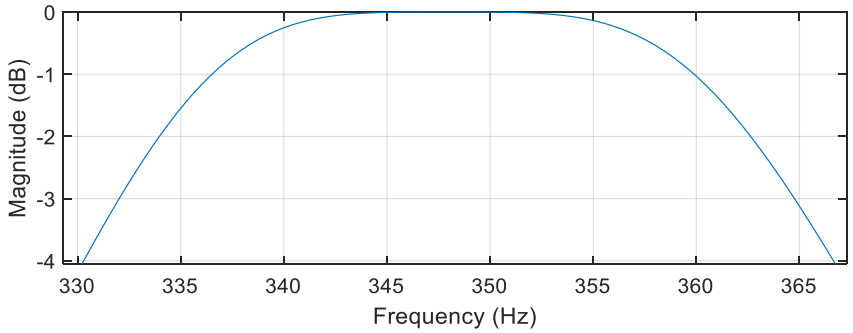
$$[f(i+1) - f(i-1)]/\alpha, \tag{2.18}$$

where $\alpha \geq 1$. In order to achieve the best performance, the value of α requires to be adjusted to tune the performance. In this thesis work, the values of α applied for two datasets are different. The bandwidth determined with $\alpha = 3$ is used for the recorded dataset; and $\alpha = 4$ is used for the

standard dataset. The magnitude frequency responses of the bandpass filter 1 in these two datasets are displayed in Figure 2.5.



(a) Bandwidth (24.5828 Hz) for $\alpha = 4$



(b) Bandwidth (32.777 Hz) for $\alpha = 3$

Figure 2.5 Magnitude frequency responses for the first bandpass filter.

2.4 Feature Extraction Procedure

To fit various network algorithms and express more precise features, there are two expressions for features, one-dimension and two-dimensions. As known in Sections 2.1, 2.2 and 2.3, there are two kinds of features separately from sparse discrete wavelet decomposition (SDWD) and bandpass filter banks (BPFB) which are based on the Mel filter banks. For both SDWD and BPFB features, the one-dimensional and two-dimensional features are all calculated. One-dimensional features are calculated as signal power values in each decomposition, while the procedure of extracting the two dimensions features is illustrated in Figure 2.6 [12].

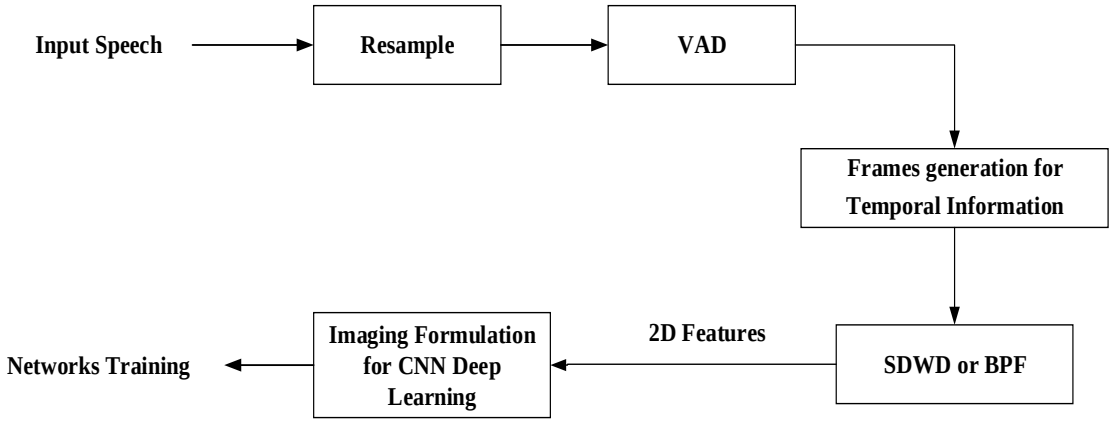


Figure 2.6 Two-dimension feature extraction and image generation.

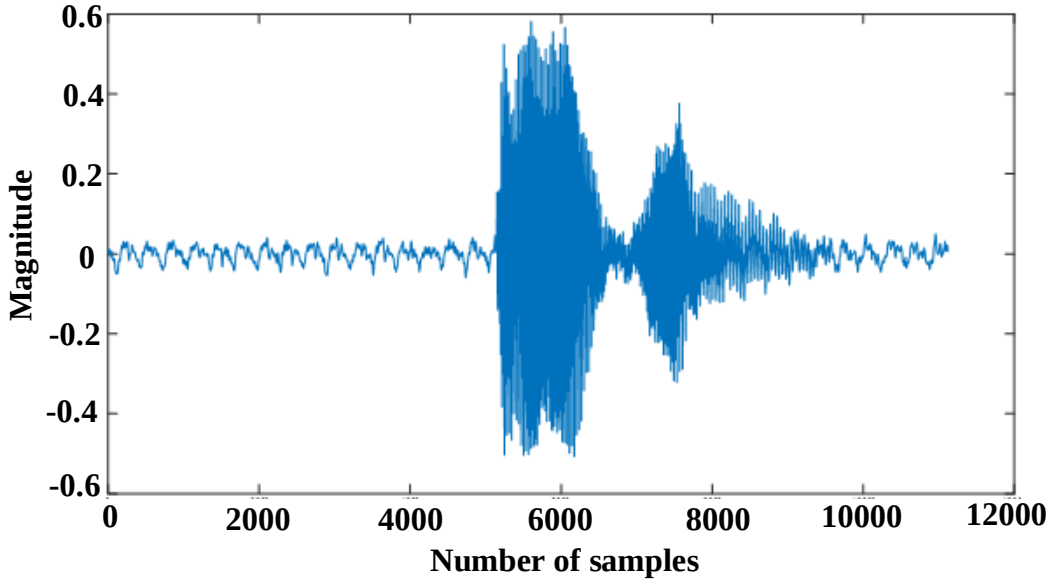
The first step is a stage of preprocessing speech, in which speech signal is subject to normalization and dc removal. The voiced segment is retrieved with a voice activity detection (VAD) algorithm. The second step is for frame generation for temporal information, in which the voice segment is divided into n ($n > 0$) frames with 50% overlapping and each frame is decomposed into 32 sub-band signal power values derived from SDWD or BPFB. Finally, the two-dimensional features, which are further formed using geometric transformation, are used as the imaging inputs for CNN. These steps will be introduced in details in Sections 2.4.1, 2.4.2 and 2.4.3, respectively.

2.4.1 Preprocessing

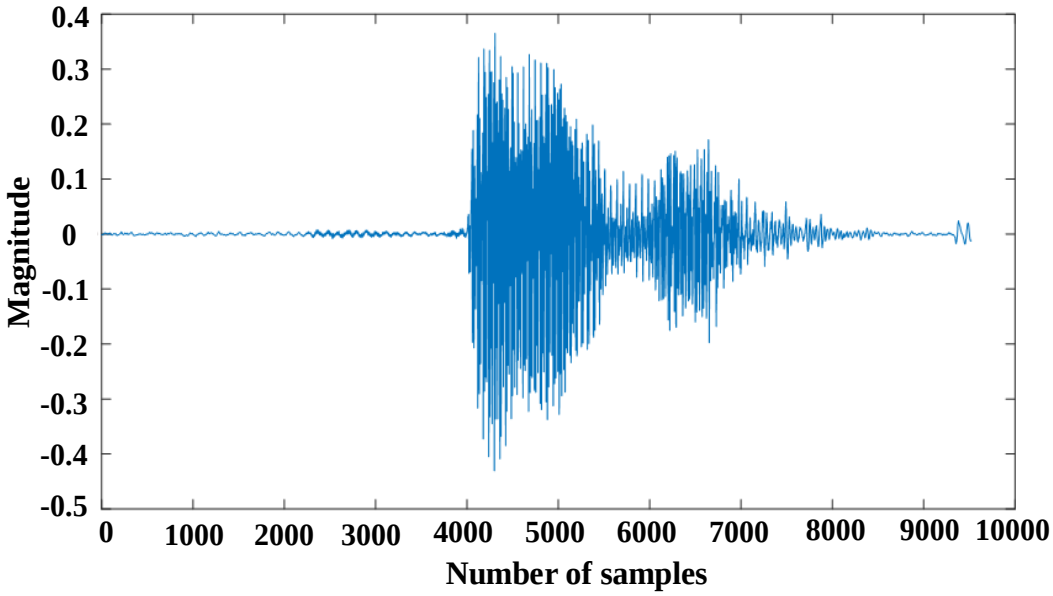
The first step is preprocessing including the resampling and voice activity detection (VAD). Because of the range of Mel scale, the sampling rate is set as 8 kHz. However, the sampling rate can be different for different datasets. Thus, resampling speech signal to 8 kHz is conducted before all steps in features extraction begin.

The second part in preprocessing is VAD. After the dc removal and normalization, the VAD is conducted to distinguish between the voiced segment and silence segment by the short-term energy (STE). With a threshold set for STE, the voiced part can be retrieved [9]. For example, as illustrated in Figure 2.7, the voice (“seven”) after dc removal and normalization doesn’t occupy the whole utterance. There are also some parts for silence. In addition, by comparing Figure 2.7 (a) and (b), it is obvious that the start and ending time for female and male voice with the same

digit are different. The female voice in (a) starts from the 5500th and ends at the 9500th sample, while the male voice starts from the 4000th and ends at the 7500th sample. This difference will also lead to the limited success for speech recognition [9].



(a) Orignal speech (“seven”, female).]



(b) Orignal speech (“seven”,male)

Figure 2.7 Original speech before VAD (digit: “seven”).

In this work, as shown in Figure 2.8 [9], the red curve over the signal (blue) is the short-term energy (STE) plot. Then the voiced segment is obtained by extracting speech signal whose STE is above the threshold value. The signal after extracting the voiced part is depicted in Figure 2.9 (a) and Figure 2.9 (b) [9].

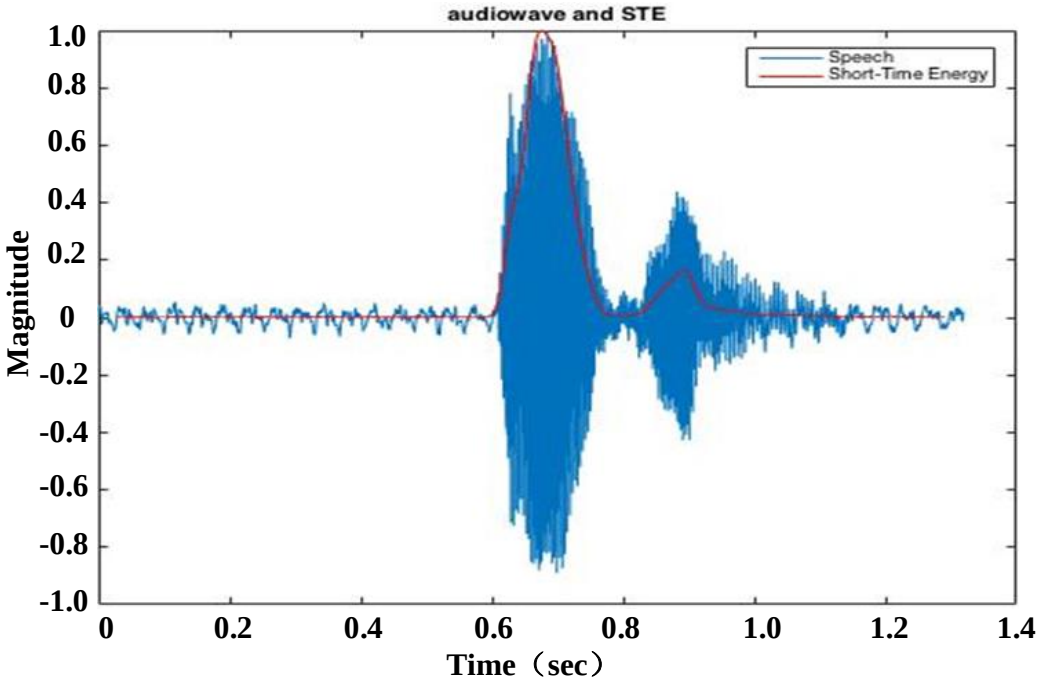
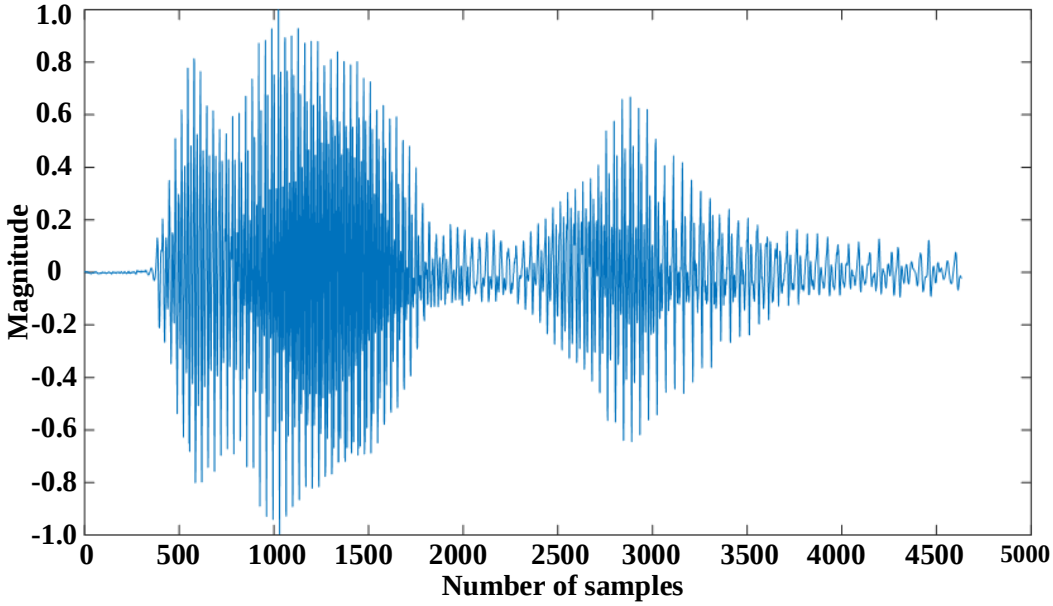
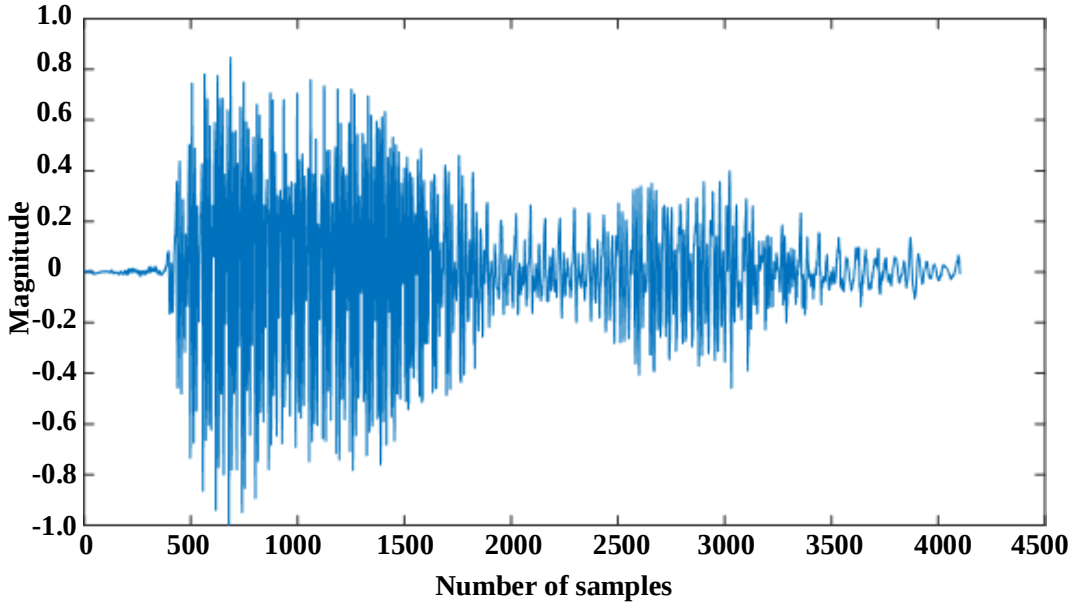


Figure 2.8 Short-term energy and normalized signal (‘seven’, female).



(a) Precocessed speech (“seven”,female),



(b) Prepreocessd speech (“seven”,male),

Figure 2.9 Prepreocessd speech (digit: “seven”).

2.4.2 Features Calculation

After the preprocessing stage, the decomposition will be implemented. The speech segment will be decomposed depending on the SWDW filters or BPBF. For the process of employing SDWD, h_0 employed in Equation (2.5) and (2.6) is an 8-tap Daubechies wavelet filter, which is

one of the typical wavelet filters whose coefficients are shown in Table 2.2. As shown in Table 2.2, the Haar wavelet is the simplest one with only two wavelet coefficients. There is no overlapping for Haar transform. Different from the Haar filters, the Daubechies filter uses overlapping windows and are smoother than the Haar filters. In addition, the Daubechies wavelet is widely applied for texture feature analysis because of compact support and orthogonal ability [9]. It usually can have better performance than Haar wavelet filters in terms of frequency response and computation load. In this thesis work, the Daubechies 8 is employed because of its compact support and orthogonal ability. Moreover, through experiments with different lengths using Daubechies 4, Daubechies 8 and Daubechies 16, the Daubechies 8 can give the high accuracy and acceptable low computation load in comparison with the other two filters [11].

Table 2.2 Typical Wavelet Filter Coefficients $h_0(k)$

Haar	Daubechies 4	Daubechies 6	Daubechies 8
0.707106781186548	0.482962913144534	0.332670552950083	0.230377813308896
0.707106781186548	0.836516303737808	0.806891509311093	0.714846570552915
	0.224143868042013	0.459877502118492	0.630880767929859
	-0.129409522551260	-0.135011020010255	-0.027983769416859
		-0.085441273882027	-0.187034811719093
		0.035226291885710	0.030841381835561
			0.032883011666885
			-0.010597401785069

For feature calculation and expression as shown in Figure 2.10, sub-bands signal power values are calculated for each frame. The power values are given by,

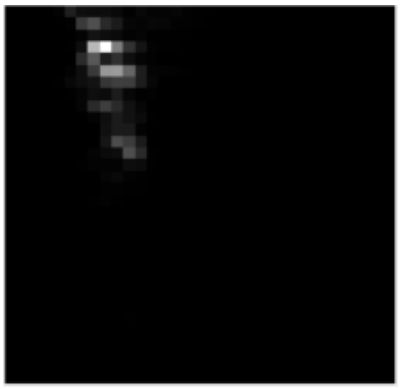
$$P_i = \sum_{j=1}^{J} E\{x_{ij}^2(k)\}, \tag{2.19}$$

where $E\{\ \}$ is the expectation operator and k denotes the sample number; $x_{ij}(k)$ is the decomposed sub-band signal for band i with J paths ($0 < J \le 3$ as shown in Table 2.1). The achieved feature vector has the following format:

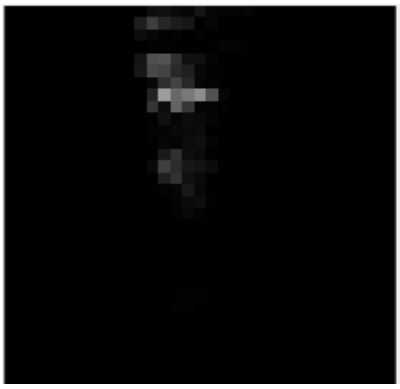
$$[P_1 \quad P_2 \quad \cdots \quad P_{32}]. \tag{2.20}$$

For feature expression, in this work, noting that 32 Mel filter banks are adopted, the one-dimensional feature consists of the power value array with a size of 1×32 for each utterance. For the two-dimensional feature, the size is $32 \times n$ ($n > 0$). The features are constructed in two axizes:

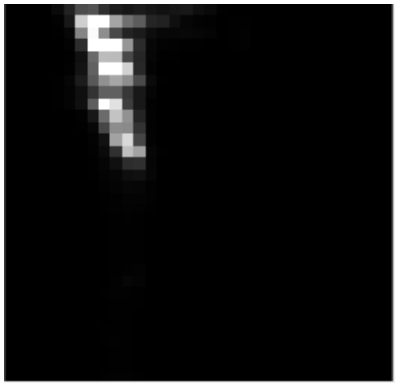
time-axis and frequency axis [12], as depicted in Figure 2.10. In the figure, the upper part values are the low-frequency components while the lower part values are the high-frequency components.



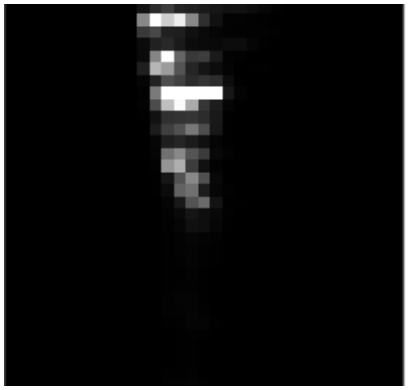
(a)SDWD 2D features (male, 32 × 33),



(b)SDWD 2D features (female, 32 × 33),



(c)BBPF 2D features (male, 32 × 33),



(d)BBPF 2D features (female, 32 × 33).

Figure 2.10 Two-dimensional features (“one”) without formulating for CNN.

There are two issues. Firstly, each of 2D features has a rectangular shape. Secondly, most feature values are concentrated in the upper left portion of the whole feature image. With this 2D feature for implementation, the CNN cannot be run since the employed CNN program requires the square size of image and CNN may degrade its performance due to the edge effect at convolution layers.

2.4.3 Imaging formulation

This section introduces the third step, imaging formulation. the goals are to transform the shape of two-dimensional features from rectangle to square; and to make the test performance

better. Based on these two goals, bilinear interpolation and geometric transformation such as the flipping operations are proposed.

Since the CNN fits its inputs in a square shape, a transformation from the rectangular image to the square image is necessary. The method proposed uses the bilinear interpolation. Before bilinear interpolation, all 2D features have the same size but rectangle shape. The details of the method are based on the summary in Reference [13].

In the implementation, the function scales an image from the original size (weight: w_o, height: h_o) to the new size (weight: w_n, height: h_n) .

Using a scaling factors of $\frac{h_o}{h_n}$ and $\frac{w_o}{w_n}$, the new pixel position (x, y) can be determined by the corresponding given original position (i, j). This transform is shown below:

$$x = i \cdot \frac{h_o}{h_n},\ y = j \cdot \frac{w_o}{w_n}. \tag{2.21}$$

If x and y are floating numbers, the rounded integers will be used as the pixel position (x', y'). Since the calculated pixel (x, y) doesn't usually exist in the scaled image, its intensity will be calculated using the bilinear interpolation with four pixels around (x, y). These four pixels are designated as (x_0, y_0), (x_0, y_1), (x_1, y_0) and (x_1, y_1). And the corresponding intensity is respectively $f(x_0, y_0)$, $f(x_0, y_1)$, $f(x_1, y_0)$ and $f(x_1, y_1)$. Note that these four pixels can be found in the original image. The horizontal interpolation is shown from (2.22) to (2.24) as

$$z_1 = \frac{f(x_1, y_0) - f(x_0, y_0)}{x_1 - x_0} \times u + f(x_0, y_0), \tag{2.22}$$

$$z_2 = \frac{f(x_1, y_1) - f(x_0, y_1)}{x_1 - x_0} \times u + f(x_0, y_1), \tag{2.23}$$

$$z = \frac{z_2 - z_1}{y_1 - y_0} \times v + z_1. \tag{2.24}$$

The vertical interpolation is shown from (2.25) to (2.27) as

$$z_1 = \frac{f(x_0, y_1) - f(x_0, y_0)}{y_1 - y_0} \times v + f(x_0, y_0), \tag{2.25}$$

$$z_2 = \frac{f(x_1, y_1) - f(x_1, y_0)}{y_1 - y_0} \times v + f(x_1, y_0), \tag{2.26}$$

$$z = \frac{z_2 - z_1}{x_1 - x_0} \times u + z_1. \tag{2.27}$$

Because of the neighborhood pixels, $x_1 - x_0 = y_1 - y_0 = 1$, the final formula can be derived as

$$Z = v \times [u \times f(x_1, y_1) + (1-u) \times f(x_0, y_1)] + (1-v) \times [u \times f(x_1, y_0) + (1-u) \times f(x_0, y_0)],$$

where u and v are respectively the decimal fractional parts of x and y. Z is the intensity of a new pixel.

After conducting the bilinear interpolation to the features in the shape like Figure 2.10, the feature sizes are all transformed to 32×32. However, by experiments, the accuracy of speech recognition using the CNN training with the 2D features is not high. Obviously, the testing performance is degraded because of the second issue in which the features reside in the upper-left region.

In theory, the image with the central information can have more distinguished results after convolutional layers so that the recognition accuracy can be increased. To “move” the information parts (power values) to the central part, the method (geometric transformation) of flipping an original two-dimensional feature is proposed. Each two-dimensional feature image is respectively flipped about the vertical, horizontal and diagonal axes. The process of flipping features can be illustrated in Figure 2.11.

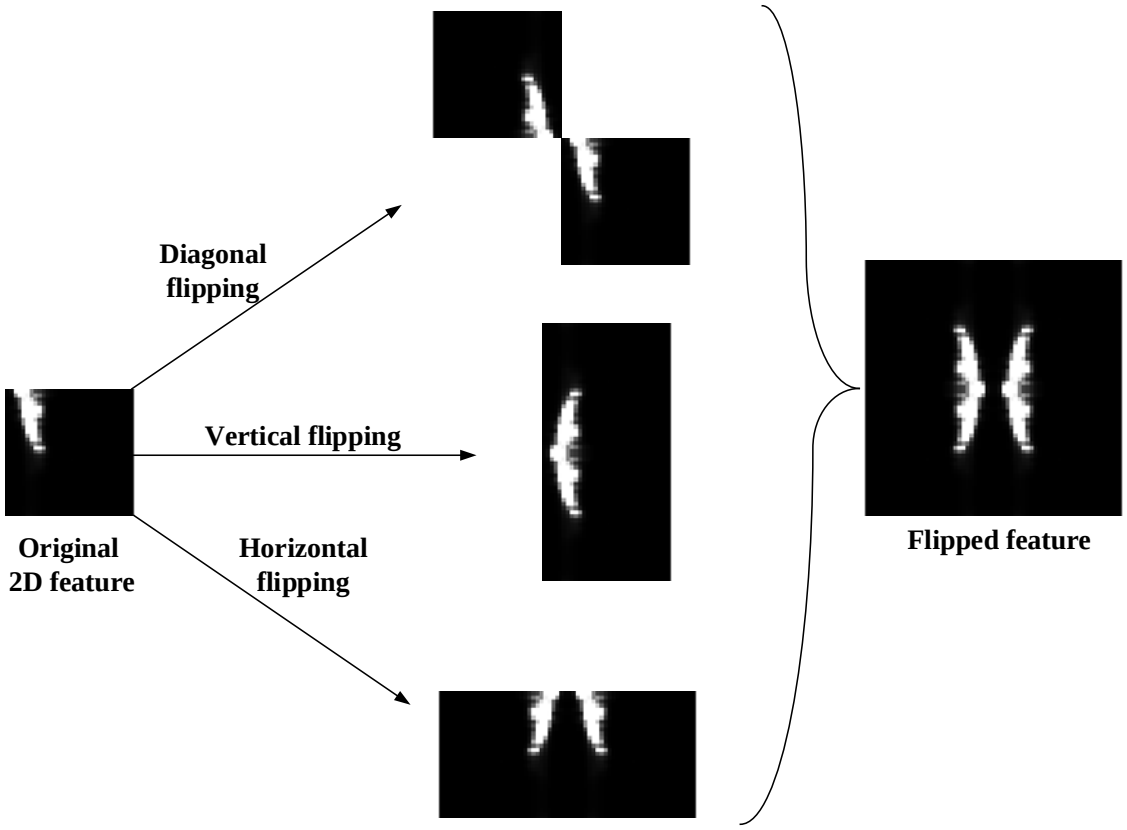


Figure 2.11 Flipping operation.

The two-dimensional features after shape-transformation and flipping are shown in Figure 2.12. Finally, all feature images now have the same size (64 × 64) with the central information. The recognition performance is significantly improved.

(a)SDWD 2D features (“one”,male, 64 × 64), (a)BBPF 2D features (“one”,male, 64 × 64).

Figure 2.12 Flipped features (digit: “one”).

CHAPTER 3. CLASSIFICATION TECHINIQUES

This chapter introduces the convolutional neural networks (CNN) algorithm and develops the techniques for training CNN using two-dimensional features calculated with sparse discrete decomposition (SDWD) and bandpass filter banks (BPFB). In order to compare the performances of CNN-SDWD and CNN-BPFB using two-dimensional features, several standard classification techniques with one-dimensional features are proposed. These techniques proposed include support vector machine (SVM), K-nearest neighbors (KNN) and random forest (RF), which are briefly reviewed in this chapter.

3.1 Support Vector Machine (SVM)

All details for Section 3.1 are based on the summary of Reference [14]. With principles of structural risk minimization (SRM), a support vector machine (SVM) was developed to solving classification problems. The SVM classification (SVC) method can also be extended to solve the regression problems, that is, support vector regression (SVR). In this thesis work, our task is to classify speech signal with ten digits. Therefore, only the applied SVC is introduced in aspects of optimal hyperplanes, feature space, and kernel function.

3.1.1 Support Vector Classification (SVC)

Begin with a two-class problem. The goal is to separate two-class data points available by a function termed “classifier”. Take Figure 3.1 as an example. Data points with two colors (two classes) scatter in the input space. There are many possible classifiers, thin orange lines in Figure 3.1, can separate data. However, there is only one classifier shown as the thick green line in Figure 3.1, which can maximize the margin (the distance from it to the nearest data point belonging to each class). Those possible classifiers are also termed as the “hyperplane”; and the one maximizing the margin is also termed as the “optimal hyperplane”.

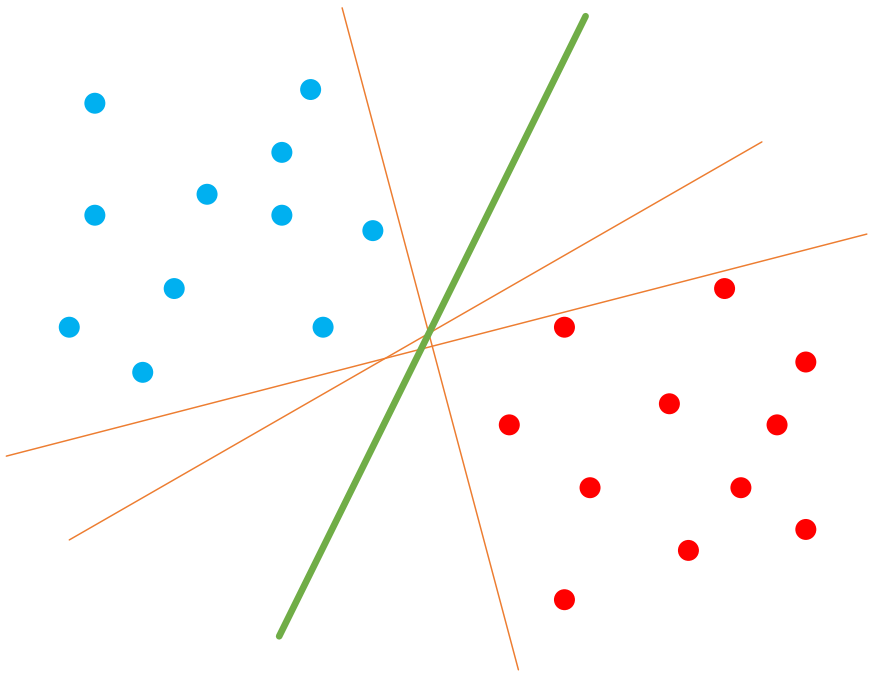


Figure 3.1 Data points and separating hyperplanes.

3.1.2 Optimal Hyperplane

Assume training data vectors belonging to two classes are given by:

$$D = \left\{(x^1, y^1), ..., (x^i, y^i), ..., (x^l, y^l)\right\}, x \in \mathbb{R}^n, y \in \{-1, 1\}, \tag{3.1}$$

where x^i ($i = 1, .., l$) is the $i\text{-}th$ training data vector with an n dimension. The value of y^i represent the class and each data belongs to (-1 or 1). The hyperplane to classify data vectors are written as:

$$\langle \omega, x \rangle + b = 0, \tag{3.2}$$

where ω is the weight vector and b is the bias. In order to optimally separate data vectors with the smallest error, that is, the maximum distance between the hyperplane and the closet vector. Equation (3.2) is constrained by:

$$\min_i \left| \langle \omega, x^i \rangle + b \right| = 1. \tag{3.3}$$

In a canonical form, the separating hyperplane must satisfy the following constrains,

$$y^i \left[\langle \omega, x^i \rangle + b \right] \geq 1, i = 1, ..., l. \tag{3.4}$$

To obtain the optimal hyperplane, the principle in words states that the norm of the weight vector is equivalent to the inverse of the distance, between the closet data point and the hyperplane. The distance $d(\omega,b,x)$ from a point x^i to a hyperplane $\langle \omega, x \rangle$ is,

$$d(\omega,b,x) = \frac{\left|\langle \omega, x^i \rangle + b\right|}{\|\omega\|}. \tag{3.5}$$

By maximizing the margin ρ in terms of the distance, the optimal hyperplane is obtained. The margin is given by,

$$\begin{aligned}
\rho(\omega,b) &= \min_{x^i, y^i=-1} (d(\omega,b,x^i)) + \min_{x^i, y^i=1} (d(\omega,b,x^i)) \\
&= \min_{x^i, y^i=-1} \left(\frac{\left|\langle \omega, x^i \rangle + b\right|}{\|\omega\|}\right) + \min_{x^i, y^i=1} \left(\frac{\left|\langle \omega, x^i \rangle + b\right|}{\|\omega\|}\right) \\
&= \frac{1}{\|\omega\|}\left(\min_{x^i, y^i=-1} \left|\langle \omega, x^i \rangle + b\right| + \min_{x^i, y^i=1} \left|\langle \omega, x^i \rangle + b\right| \right) \\
&= \frac{2}{\|\omega\|}.
\end{aligned} \tag{3.6}$$

Thus, the optimal separating hyperplane can minimize

$$\Phi(\omega) = \frac{1}{2}\|\omega\|^2. \tag{3.7}$$

The solution to optimal hyperplane by minimizing Equation (3.7) under the constrains in (3.4) is introduced with the Lagrange multipliers shown as in (3.8).

$$\Phi(\omega,b,\alpha) = \frac{1}{2}\|\omega\|^2 - \sum_{i=1}^{l} \alpha_i \left(y^i \left[\langle \omega, x^i \rangle + b \right] - 1 \right), \tag{3.8}$$

where α_i notes a set of Lagrange multipliers. The dual problem can be further introduced that (3.8) must be minimized with respect to ω, b and be maximized with respect to α ($\alpha \geq 0$). The dual problem is shown as

$$\max_{\alpha} W(\alpha) = \max_{\alpha} \left(\min_{\omega,b} \Phi(\omega,b,\alpha) \right). \tag{3.9}$$

Taking the partial derivatives of Φ to ω and b, respectively, setting them to zero lead to:

$$\begin{aligned}\frac{\partial \Phi}{\partial b} = 0 \Rightarrow \sum_{i=1}^{l} \alpha_i y_i = 0, \\ \frac{\partial \Phi}{\partial \omega} = 0 \Rightarrow \omega = \sum_{i=1}^{l} \alpha_i y_i x_i .\end{aligned} \tag{3.10}$$

Substituting (3.10) into (3.8), the dual problem can be given in further by,

$$\max_{\alpha} W(\alpha) = \max_{\alpha} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \left\langle x_i, x_j \right\rangle + \sum_{k=1}^{l} \alpha_k , \tag{3.11}$$

and the solution with corresponding constrains to the dual problem is given by,

$$\begin{aligned}\alpha^* = \arg\min_{\alpha} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \left\langle x_i, x_j \right\rangle - \sum_{k=1}^{l} \alpha_k , \\ \alpha_i \geq 0 \ \ (i = 1, ..., l), \sum_{j=1}^{l} \alpha_i y_i = 0.\end{aligned} \tag{3.12}$$

Then the optimal hyperplane can be determined by the optimal weight vectors and bias.

$$\begin{aligned}\omega^* = \sum_{i=1}^{l} \alpha_i y_i x_i , \\ b^* = -\frac{1}{2}\left\langle \omega^*, x_r + x_s \right\rangle ,\end{aligned} \tag{3.13}$$

where x_r and x_s are any support vectors which belong to their classes satisfying the constrains given by,

$$\alpha_r, \alpha_s > 0, y_r = -1, y_s = 1. \tag{3.14}$$

As the solution of dual problem from (3.8) to (3.14), the optimal hyperplane is shown in Figure 3.2.

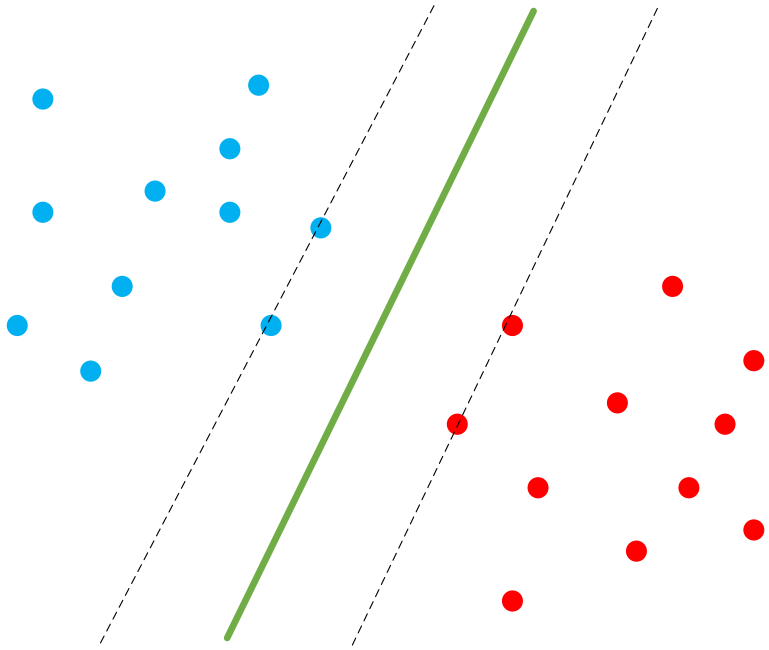


Figure 3.2 Optimal Hyperplane.

The classifier is then given as

$$f(x) = \operatorname{sgn}\left(\left\langle \omega^*, x \right\rangle + b^*\right).$$

3.1.3 Generalized Optimal Hyperplane

Section 3.1.1 introduces the case that data points are linearly separable. However, in many practical classification problems, the data points can't be linearly separable but can be nonlinearly separable. There are two factors lead to the case. One is that some individual noised data added to the data points which are linearly separable as shown in Figure 3.3. Another is due to the characteristic of distribution of data points themselves, which will be introduced in Section 3.1.3.

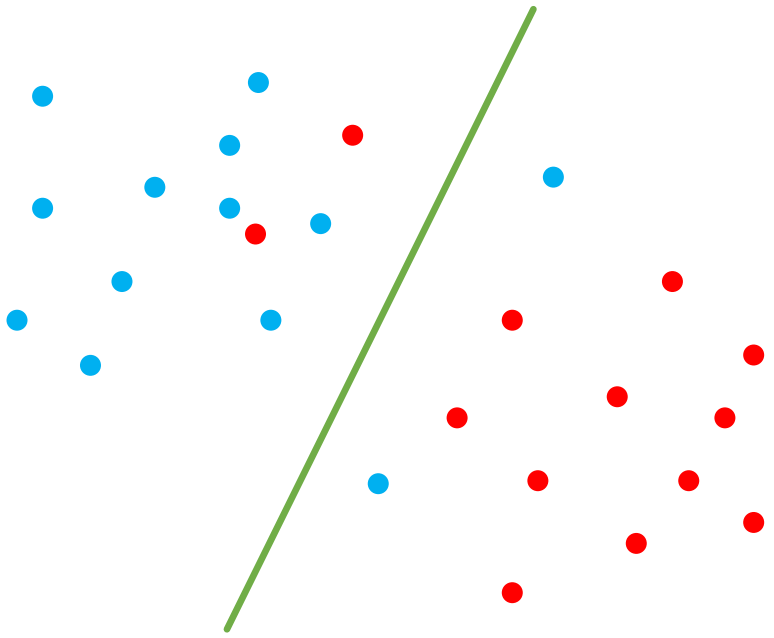


Figure 3.3 Data points with noise.

In order to enable the optimal hyperplane to generalize the data points in Figure 3.3, a penalty function is introduced by considering those noised data points. The penalty function is given by,

$$F_\sigma(\xi) = \sum_i \xi_i^\sigma, \sigma > 0, \tag{3.15}$$

where ξ_i are the misclassification errors. Not all of ξ_i are nonzero. The values depend on how is the significance of the misclassification of each data point. With the consideration for misclassification, the constrains in (3.4) can be modified as,

$$y^i\left[\left\langle \omega, x^i \right\rangle + b\right] \geq 1 - \xi_i, i = 1, ..., l, \tag{3.16}$$

where $\xi_i \geq 0$. And the hyperplane is derived by minimizing the function,

$$\Phi(\omega) = \frac{1}{2}\|\omega\|^2 + C\sum_i \xi_i, \tag{3.17}$$

where C is given and subject to (3.16). The solution with the Lagrange multipliers is given by,

$$\Phi(\omega, b, \alpha, \xi, \beta) = \frac{1}{2}\|\omega\|^2 - C\sum_i \xi_i + \sum_{i=1}^{l} \alpha_i \left(y^i \left[\left\langle \omega, x^i \right\rangle + b\right] - 1 + \xi_i \right) - \sum_{j=1}^{l} \beta_i \xi_i, \tag{3.18}$$

where α and β are both Lagrange multipliers. Solving the dual problem yields (3.19) below:

$$\begin{aligned}
&\frac{\partial \Phi}{\partial b} = 0 \Rightarrow \sum_{i=1}^{l} \alpha_i y_i = 0, \\
&\frac{\partial \Phi}{\partial \omega} = 0 \Rightarrow \omega = \sum_{i=1}^{l} \alpha_i y_i x_i, \\
&\frac{\partial \Phi}{\partial \zeta} = 0 \Rightarrow \alpha_i + \beta_i = C.
\end{aligned} \tag{3.19}$$

Similar to Section 3.1.2, the optimal α can be derived as,

$$\begin{aligned}
&\alpha^* = \arg\min_{\alpha} \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \left\langle x_i, x_j \right\rangle - \sum_{k=1}^{l} \alpha_k, \\
&0 \le \alpha_i \le C (i = 1, ..., l), \sum_{j=1}^{l} \alpha_i y_i = 0.
\end{aligned} \tag{3.20}$$

3.1.4 Feature Space

To tackle with the nonlinear separable problem due to the characteristic of distribution of data points themselves, the approach mapping the input space into a high dimensional feature space is employed. To do so, the kernel function is adopted. The process of nonlinear mapping is illustrated in Figure 3.4.

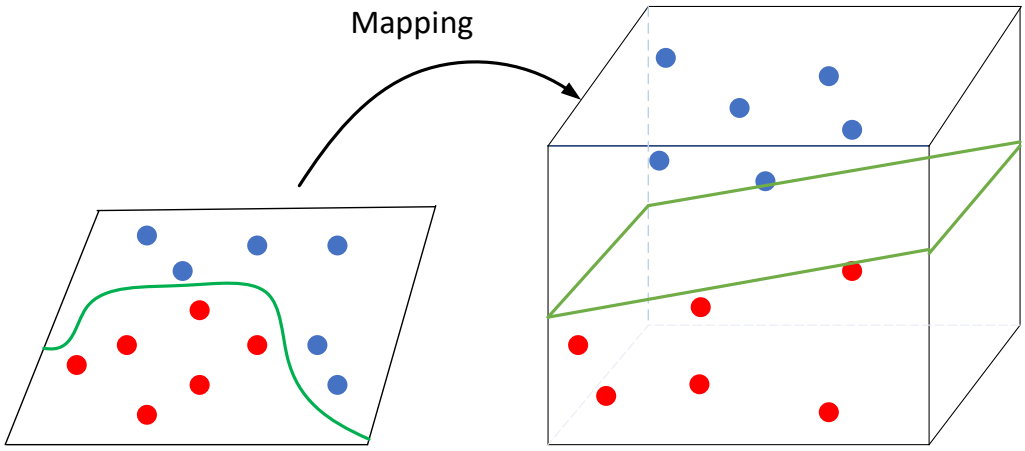


Figure 3.4 Mapping from input space to feature space.

The mapping from input space x to feature space $\phi(x)$ is given by $x \Rightarrow \phi(x)$. And the inner product in the feature space is equal to the kernel in the input space as shown in (3.21).

$$K(x,x') = \left\langle \phi(x), \phi(x') \right\rangle = \sum_{m}^{\infty} \alpha_m \phi_m(x) \phi_m(x'), \alpha_m \geq 0. \tag{3.21}$$

Thus, in the feature space, the process of deriving the optimal hyperplane is the same as the one for linearly separable data points [from (3.8) to (3.14)]. Optimal α, ω and b are derived as

$$\begin{aligned} &\alpha^* = \arg\min_{\alpha} \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K \left\langle x_i, x_j \right\rangle - \sum_{k=1}^{l} \alpha_k, \\ &0 \leq \alpha_i \leq C (i = 1, ..., l), \sum_{j=1}^{l} \alpha_i y_i = 0. \end{aligned} \tag{3.22}$$

$$\begin{aligned} &\left\langle \omega^*, x \right\rangle = \sum_{i=1}^{l} \alpha_i y_i K(x_i, x), \\ &b^* = -\frac{1}{2} \sum_{i=1}^{l} \alpha_i y_i \left[K(x_i, x_r) + K(x_i, x_s) \right]. \end{aligned} \tag{3.23}$$

The classifier becomes

$$f(x) = \operatorname{sgn}\left(\left\langle \omega^*, x \right\rangle + b^* \right). \tag{3.24}$$

There are many kernel functions available. The most common method is the polynomial mapping which is given by,

$$K(x,x') = \left(\left\langle x, x' \right\rangle \right)^d \text{ or } K(x,x') = \left(\left\langle x, x' \right\rangle + 1 \right)^d, \tag{3.25}$$

when order $d = 1$, (3.25) is a linear kernel function; and when $d = 2$, (3.25) is a quadratic kernel function. If the covariance matrix is diagonal, the linear or quadratic kernel function is termed as diaglinear or diagquadratic kernel functions.

There are also other functions such as Gaussian kernel function, spline kernel function and so on. In this thesis work, the polynomial kernel function can achieve the best performance.

3.1.5 Multiclass SVM

The SVM itself is a two-class classifier. However, in many practical problems, classifiers with more than two classes are necessary. The widely applied approach is aggregating two-class classifiers, defined as single multiclass classifier, into multiple binary classifiers. The method proposed in the thesis is one-versus-all method.

One-versus-all method is used for distinguishing between one label and other labels. In this thesis work, ten two-class classifiers need to be trained. Take an example. When the i-th sub-

classifier is training, the samples belonging to the $i\text{-}th$ class are labeled as positives while other samples are labeled as negatives. In the testing phase, all classifiers are applied to a test sample with unknown class and predict its label which is the most common among all classifier outputs. The function is given by

$$\hat{y} = \arg \max_{k \in \{1,\ldots,K\}} f_k(x), \tag{3.26}$$

where f_k are classifiers for $k \in \{1,\ldots,k\}$. The accuracy rates for multiclassification svm systems are shown in Table 3.1. The "Order" in Table 3.1 represents the order of kernel function. For example, if order is 1, the kernel function is exactly the linear function; if order is 2, the quadratic function is the kernel function.

Table 3.1 Multisvm accuracy

Table A Standard Datasets

Accuracy / Order	Mean	
	SDWD	BPFB
1	62.50%	65.00%
2	52.50%	57.50%
3	62.50%	67.50%
4	55.00%	57.50%
5	50.00%	62.50%
6	45.00%	50.00%

Table B Recorded Datasets

Accuracy / Order	Mean	
	SDWD	BPFB
1	87.50%	87.50%
2	87.50%	92.50%
3	92.50%	90.00%
4	85.00%	82.50%
5	85.00%	80.00%
6	62.50%	70.00%

3.2 Random Forest

Based on the idea of ensemble learning, random forest (RF) is an algorithm that constructs multiple decision trees. In order to classify a new object from an input vector, the vector is put from up to down in each tree. Each tree is a classifier and can give a result of classification, which can be taken as a "vote" for the resultant class. Finally, the whole forest chooses the classification with the most votes. Each tree is grown as the principles briefly described below [15],

a. If there are N training samples given, each tree chooses N samples at random with replacement from the original training samples. This principle is based on bootstrap sampling method;

b. If each training vector contains M variables (features), m variables are selected randomly, where m ($m < M$) is a specified constant integer value. The best split on these m is used to split the node;

c. Each tree is grown as large as possible to its extent and has no pruning.

3.2.1 Decision Tree

The decision tree is a classifier using tree structure with internal nodes, leaf nodes and path. Each internal node can specify the test on single variable (feature). Each leaf node indicate the output class. And a path is the disjunction of the test to make the final decision.

The algorithm for decision tree construction is usually conducted from up to down, which is from the root of tree down to the leaf node, by choosing one of variables which can best split the nodes [16]. In order to measure the "best" splitting, the approach using the information gain, which measures the homogeneity of the response values within m input variables is considered. The information gain is used to decide which variable is split at each step in the process of constructing the tree. In each step, the variable with the highest information gain among m variables should be selected for splitting.

3.2.2 Random Forest Algorithm

The general technique of bootstrap aggregating or bagging is applied to train the algorithm for random forests.

In this thesis work, the tree bagging is employed. The process of bagging is shown as below [17].

Given the training set as $X = x_1, ..., x_N$ with the assigned classes as $Y = y_1, ..., y_N$, bagging can repeatedly select a random sample with replacement from a training set to these samples for N times.

1. Choosing samples with replacement from N training samples. The samples selected are called as X_r and Y_r where $r = 1, ..., R$;
2. Constructing each decision tree f_T on X_r and Y_r.

After the forest is completely constructed, the new sample x' can be assigned to a class by each tree f_T. Finally, x' is assigned to the class with the most votes. In this work, the accuracy versus number of trees in two datasets are shown in Table 3.2.

Table 3.2 RF accuracy Vs Trees number (10 independent runs)

Table A Standard Datasets

Methods / Trees number	Mean		Standard Deviation	
	SDWD	BPFB	SDWD	BPFB
10	65.00%	80.50%	0.0441	0.0307
11	68.25%	80.00%	0.0481	0.0289
12	68.50%	80.00%	0.0444	0.0354
13	68.75%	81.50%	0.0429	0.0412
14	70.00%	82.50%	0.0391	0.0408
15	68.75%	81.75%	0.0475	0.0374
16	69.50%	83.75%	0.0369	0.0429
17	69.25%	83.75%	0.0374	0.0556
18	69.75%	86.00%	0.0432	0.0175
19	70.00%	83.25%	0.0667	0.0334
20	72.00%	83.75%	0.0483	0.0377

Table 3.2 Continued.

Table B Recorded Datasets

Methods / Trees number	Mean		Standard Deviation	
	SDWD	BPFB	SDWD	BPFB
10	95.50%	90.25%	0.0387	0.0661
11	95.25%	90.75%	0.0322	0.0290
12	96.75%	91.25%	0.0290	0.0317
13	96.75%	91.50%	0.0290	0.0428
14	97.25%	92.00%	0.0219	0.0230
15	97.75%	92.00%	0.0142	0.0387
16	98.00%	92.75%	0.0197	0.0275
17	98.25%	91.75%	0.0169	0.0457
18	98.00%	92.25%	0.0158	0.0299
19	98.75%	93.00%	0.0177	0.0284
20	97.75%	93.25%	0.0219	0.0237

3.3 K Nearest Neighbors

In the field of pattern recognition, the k nearest neighbors (KNN) algorithm is a method for classification and regression [18]. It is a simple one of machine learning algorithms. It can be interpreted as a kind of learning algorithm based on instance. The output for prediction depends on the k closest training samples for regression or classification. In our thesis work, the KNN classification is required and will be introduced in this section.

For KNN classification, an output is classified depending on a majority vote of its neighbors. The output, classified object, is assigned to the class as the most common among its nearest neighbors [18]. The number of nearest neighbors is set previously and a typically small positive integer. For the special case when $k = 1$, the output is simply classified to the class that the single neighbor is assigned to.

3.3.1 KNN Algorithm

In this thesis work, the samples are vectors in the feature space. They consist of feature data and corresponding class labels. The number of neighbors nearest to the new sample, k, is selected as an integer value. In the classification phase, all test samples are unlabeled vector and each test sample will be assigned to its predicted label according to most frequent training labels among k training samples which are the closet to that test sample [19].

For measuring distance between one of neighbors and test sample point, the Euclidean distance is widely applied. In Euclidean n-space, the distance between p and q is given by

$$d(p,q)=\sqrt{(q_1-p_1)^2+(q_2-p_2)^2+(q_3-p_3)^2+...+(q_n-p_n)^2}=\sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}, \qquad (3.27)$$

where $p=(p_1,p_2,p_3,...,p_n)$ and $q=(q_1,q_2,q_3,...,q_n)$.

The algorithm given from (3. 27) is based on Reference [20].

Another method, spearman correlation [21], can achieve better performance in some cases. The algorithm is given by

(1) Determining the training samples and testing samples;

(2) Calculating the spearman correlation coefficient given by,

$$d_{ij}=1-\frac{6\sum_{i=1}^{n}(rank(x_i)-rank(t_j))^2}{n(n^2-1)}, \qquad (3.28)$$

where x_i is the i-th training sample; and the t_j is the j-th testing sample.

(1) Listing the d_{ij} in order and choose the k minimum ones (k neighbors);

(2) Finding classes these k neighbor belonging to. The class with the majority is the one the test sample should be assigned to.

3.3.2 Prediction and Forecasting

In training samples $X=(x_1,x_2,...,x_n)$ and its corresponding response Y. Depending on practical classification problems, Y can be categories or numbers. The “categories” means the values of Y are different signs or characters; and in this work, the values of Y are numbers the same as the digits classified. A simple example with two-classification is present below.

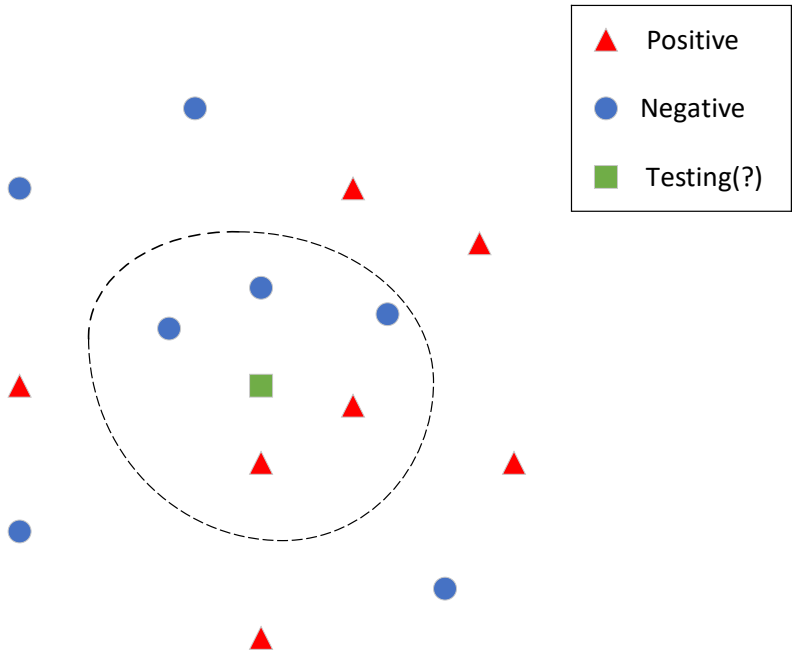


Figure 3.5 Two-classification (“categories”).

The Figure 3.5 shows the two-class classification with the response for “categories” (“positive” and “negative”). If the k is set as 5, the five training samples which are nearest to the testing sample will be chosen. Within the circled region as indicated in Figure 3.5, there are three negative samples and two positive samples. Thus, the testing sample belongs to the “negative” class.

In this thesis work, there are standard data set and recorded data set each containing ten digits. The accuracy rates are shown in Table 3.3. The spearman correlation is applied to features for both standard-BPFB (BPFB features from standard dataset) and standard-SDWD as well as recorded-BPFB. The Euclidean distance is applied to the recorded-SDWD.

Table 3.3 KNN accuracy Vs Neighbors

Table A Standard Datasets

Accuracy / Neighbors	SDWD	BPFB
1	70.00%	87.50%
2	67.50%	85.00%
3	72.50%	82.50%
4	62.50%	80.00%
5	60.00%	75.00%
6	62.50%	72.50%
7	60.00%	72.50%
8	60.00%	67.50%
9	60.00%	67.50%
10	65.00%	65.00%
11	62.50%	62.50%
12	60.00%	67.50%
13	60.00%	65.00%
14	62.50%	65.00%
15	62.50%	65.00%
16	60.00%	57.50%
17	57.50%	57.50%
18	55.00%	57.50%
19	55.00%	60.00%
20	55.00%	60.00%

Table 3.3 Continued.

Table B Recorded Datasets

Accuracy / Neighbors	SDWD	BPFB
1	92.50%	95.00%
2	87.50%	90.00%
3	92.50%	92.50%
4	90.00%	92.50%
5	90.00%	92.50%
6	92.50%	90.00%
7	87.50%	87.50%
8	87.50%	85.00%
9	87.50%	82.50%
10	80.00%	87.50%
11	82.50%	75.00%
12	77.50%	65.00%
13	77.50%	65.00%
14	77.50%	62.50%
15	80.00%	57.50%
16	77.50%	60.00%
17	77.50%	52.50%
18	72.50%	55.00%
19	72.50%	50.00%
20	72.50%	55.00%

3.4 Convolutional Neural Networks (CNN)

The convolutional neural network (CNN) is one of deep neural networks. It is widely applied for analyzing visual imaging, such as image and video recognition, natural language processing [22] and recommender systems [23]. Convolutional neural networks take inputs being made from images and are sensitive to the architecture.

In this thesis work, the procedure how CNN wroks can be illustrated in Figure 3.6. The first phase is the training phase including forward propagation and back propagation; and the second phase is the testing pahse. In the training process, because of the batch size is set as two (2), every time two of training images are taken as the inputs to CNN. There are a total number of 280 trainig images. Each training epoch contains one hundred and forty (140) taining times. After the CNN model has been trained for m epochs, forty (40) testing images are applied to the model to obtain

the testing outputs. Finally, the testing labels given are compared with the testing outputs such that the testing error can be obtained.

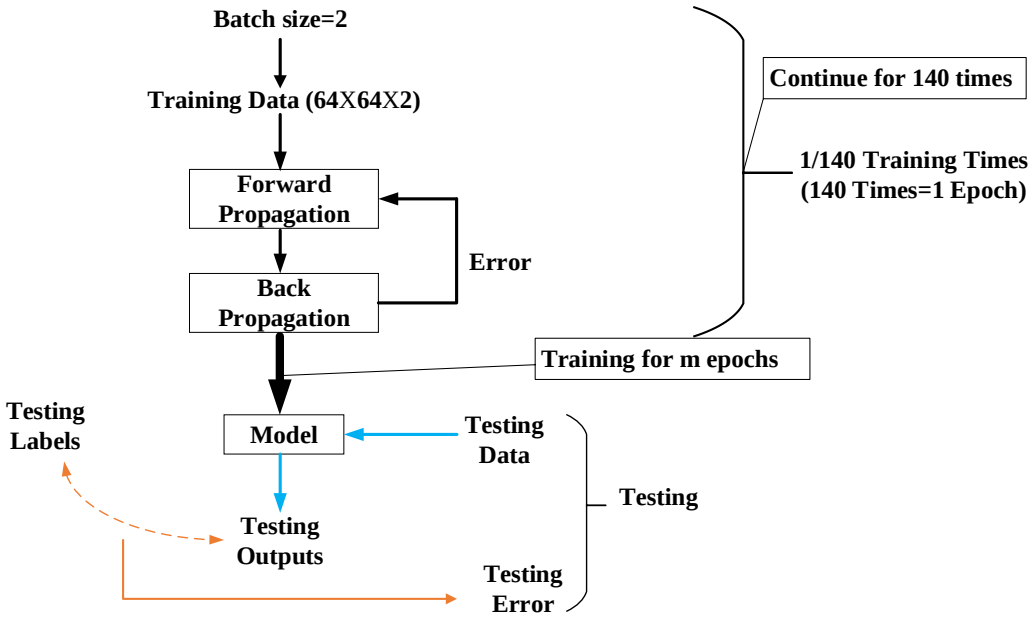


Figure 3.6 CNN function procedure.

The CNN structure, spatial arrangement, forward and back propagation are introduced in Section 3.4.1, 3.4.2, 3.4.3 and 3.4.4, respectively.

3.4.1 Convolutional Neural Network (CNN) Structure

Typically, a convolutional neural network consists of an input layer, an output layer and many hidden layers between input and output layers. And the hidden layers consist of convolutional layers, pooling layers, activation function, fully connected layers and normalization layers [24]. In this work, no normalization layers are employed. There are some parameters set in the layer i: kernel filter K_i (the matrix with random numbers initially) and bias array b_i for each convolutional layer; the mean operation and the subsample rate of two (2) for each pooling layer; the weights matrix W_i and bias array B_i (to distinguish the bias array in convolutional layer) for each fully-connected layer. Note that the size and the number of kernel filters in each convolutional layer should be considered. Typically, the pooling layer is periodically inserted between successive

convolutional layers in CNN architecture. Pooling layers can effectively reduce the representation size, parameters amount and computation load. In fact, the pooling layers are used for controlling overfitting [24]. The same as convolutional layers, the pooling layer operates independently on each image. The most common form is a pooling layer with each filter with a size of 2×2 and 2 down-samples to every image.

In the implementation, the forward propagation is illustrated in the Figure 3.7, where the structure consists of two successive pairs of convolutional layers and pooling layers as well as three fully-connected layers.

As shown in Figure 3.7, when one input image is inputted to the CNN structure, it is convoluted by each of eleven (11) kernel filters each with a size of 17×17 and then eleven (11) convolutional feature maps with a size of 48×48 are generated. Then each of feature maps is operated by the next pooling layer using the mean operation, resulting in the size of each operated map as 24×24. Next, each feature map in the mean pooling layer 1 is convoluted by twenty-two (22) kernel filters each with a size of 5×5 and two hundred and twenty-four (242) feature maps each with a size of 20×20 are obtained. Then these output maps are all conducted mean operation in the mean pooling layer 2 and each of 242 operated maps from the layer has a size of 10×10. Finally, all maps are processed by three fully-connected layers with one hundred and fifty (150), fifty (50) and ten (10) nodes in order.

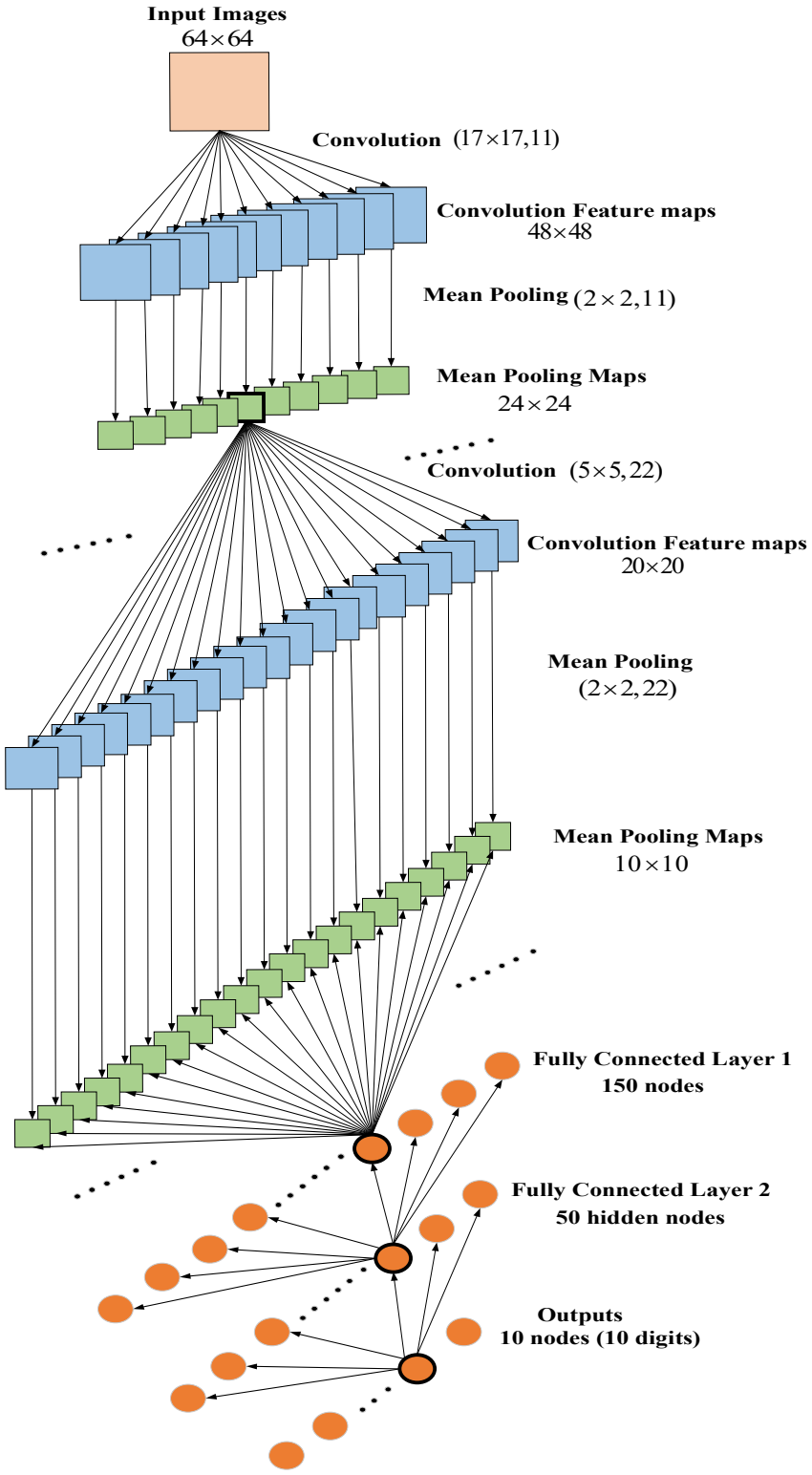


Figure 3.7 CNN structure.

3.4.2 Spatial Arrangement

The spatial arrangement is about a process of the size change from each input map to output map of each convolutional or pooling layer. The size change is controlled by three parameters: numbers of filters, stride and zero-padding.

The stride is specified as steps which each filter is sliced. For example, when the stride is 1, each filter will be moved for one pixel at a time. Thus, if the stride value is more than 1, this will produce smaller output volume. The zero-padding means the size of padding zeros around the border of each input maps [24].

In summry, the process of a convolutional or pooling layers is shown below:.

1. Accepting each of input maps as $W_1 \times H_1$ from the previous layer $(i-1)$;
2. Parameters required

   F : size of kernel filters,

   S : number of strides,

   P : number od padding zeros;
3. After the layer i, the output size is $W_2 \times H_2$, where

   $W_2 = \frac{W_1 - F + 2P}{S} + 1$ (The width of outputs),

   $H_2 = \frac{H_1 - F + 2P}{S} + 1$ (The height of outputs);
4. For parameters sharing, there are $F \times F$ weights for each filter. In this work, the initial weight matrix for each convolution layer is initialized with random numbers and the bias array is initialized to zeros; for the pooling layer, the 2×2 kernel filters contain all elements equal to 0.25. Conducting the convolution of the kernel filter with each of feature maps is ensentially a mean operation. The output maps of each pooling layer are obtained by conducting subsampling to the convolutional results with the rate of two (2).

3.4.3 Forward Propagation

Before the forward propagation, initializing CNN structure is conducted firstly for corresponding kenrnel filters, weights matrics and bias arries. The forward propagation algorithms for the convolutional layer, the pooling layer and the fully-connected layer are illustrated below.

The forward propagation in the convolutional layer consists of the calculating the sum of convolutiona which is described by,

$$C(kk) = f_i\left(conv\left((i-1).featuremaps(j), K_i(kk)\right) + b_i\right), \tag{3.29}$$

for kk=1,2,...,KK, where $conv$ represents the convolution, j is the index of the training time (the input volume of $64\times 64\times 2$ is trained each time), $K_i(kk)$ represents that the $kk\text{-}th$ kernel filter in the layer i and KK is the number of kernel filters in the layer i, $C(kk)$ is the $kk\text{-}th$ output from the layer i, f_i is the activiation function in the layer i and b_i represents the bias array in the layer i.

For the pooling layer, the kernel filter for mean operation is expressed as

$$h = \begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix}.$$

And the outputs for forward propagation is given by

$$P(kk) = conv\left((i-1).featuremaps(j), (i).h(kk)\right), \tag{3.30}$$

for kk=1,2,...,KK, where $conv$ represents the convolution, j is the index of the training time, $(i).h(kk)$ represents that the $kk\text{-}th$ kernel filter in the layer i and KK is the number of kernel filters in the layer i and $P(kk)$ is the $kk\text{-}th$ output from the layer i. Then conducting subsample operation to $P(kk)$ and each output map of the pooling layer is obtained.

For the fully-connected layer, the process is shown as

$$F(kk) = f_i\left(W_i \cdot \left((i-1).outputs\right) + B_i\right), \tag{3.31}$$

where $(i).B$ represents the bias array in the layer i, the W_i is the weights matrix in the layer i, $(i-1).outputs$ represents the outputs in the layer $(i-1)$ and f_i is the activiation function in the layer i.

3.4.4 Back Propagation

The CNN network is trained using the backpropogation algorithm. In the backpropogation, the parameters of kernel filters, weight matrices and bias arraies at each layer are updated using their gradients via the gradient descent algorithm from back to start. The backpropogation

algorithm minimizes the loss function such as the total squared errors (the error is the difference between the predicted labels and true training labels) at the last output layer with the specifications of activation function, learning rate, and the CNN structure. The CNN structures used in this work are presented in the next section.

3.4.5 Conclusion and Implementation

Based on Section 3.4.1 to 3.4.4, all parameters including kernel filters, weights matrix, bias arraies and error are adjusted by many training times. The number of training times is $140 \times m$, where m is the number of epochs. When the training phase is done, the CNN model is determined and the testing data is taken as inputs to the model. By comparing testing outputs and testing labels given, the testing error can be obtained by the precent of testing outputs not equal to the labels over the number of the whole testing labels (40).

There are four implementation structures in the experiments, that is: standard-SDWD (standard data set), standard-BPFB (standard data set), recorded-SDWD (recorded data set) and recorded-BPFB (recorded data set), which are are expressed as:

Standard-SDWD:

$[I_{1,280}, C_{2,11}(\mathrm{Re}\,lu), P_{3,11}(mean), C_{4,22}(\tanh), P_{4,22}(mean), F_{5,150}(sigm), F_{6,50}(\tanh), F_{7,10}(sigm)]$;

Standard-BPFB:

$[I_{1,280}, C_{2,10}(\mathrm{Re}\,lu), P_{3,10}(mean), C_{4,20}(\tanh), P_{4,20}(mean), F_{5,150}(sigm), F_{6,50}(\tanh), F_{7,10}(sigm)]$;

Recorded-SDWD:

$[I_{1,280}, C_{2,10}(\mathrm{Re}\,lu), P_{3,10}(mean), C_{4,20}(\tanh), P_{4,20}(mean), F_{5,150}(sigm), F_{6,50}(tanh), F_{7,10}(sigm)]$;

Recorded-BPFB:

$[I_{1,280}, C_{2,10}(\mathrm{Re}\,lu), P_{3,10}(mean), C_{4,20}(\tanh), P_{4,20}(mean), F_{5,150}(sigm), F_{6,50}(\tanh), F_{7,10}(sigm)]$,

where $I_{1,280}$ represents the 280 input maps at the input layer (the first layer); and notations within the parenthesis are all the activation functions in each layer. For example, $F_{5,150}(sigm)$ means the layer 5 is the fully-connected layer with 150 nodes and activation function is the sigmoid function.

CHAPTER 4. RECOGNITION PERFORMANCE

The section presents all speech recognition results from one-dimensional (1D) and two-dimensional (2D) features using these network algorithms. The support vector machine (SVM), random forest (RF), and k nearest neighbors (KNN) use the 1D features while the convolutional neural network (CNN) adopts the 2D features.

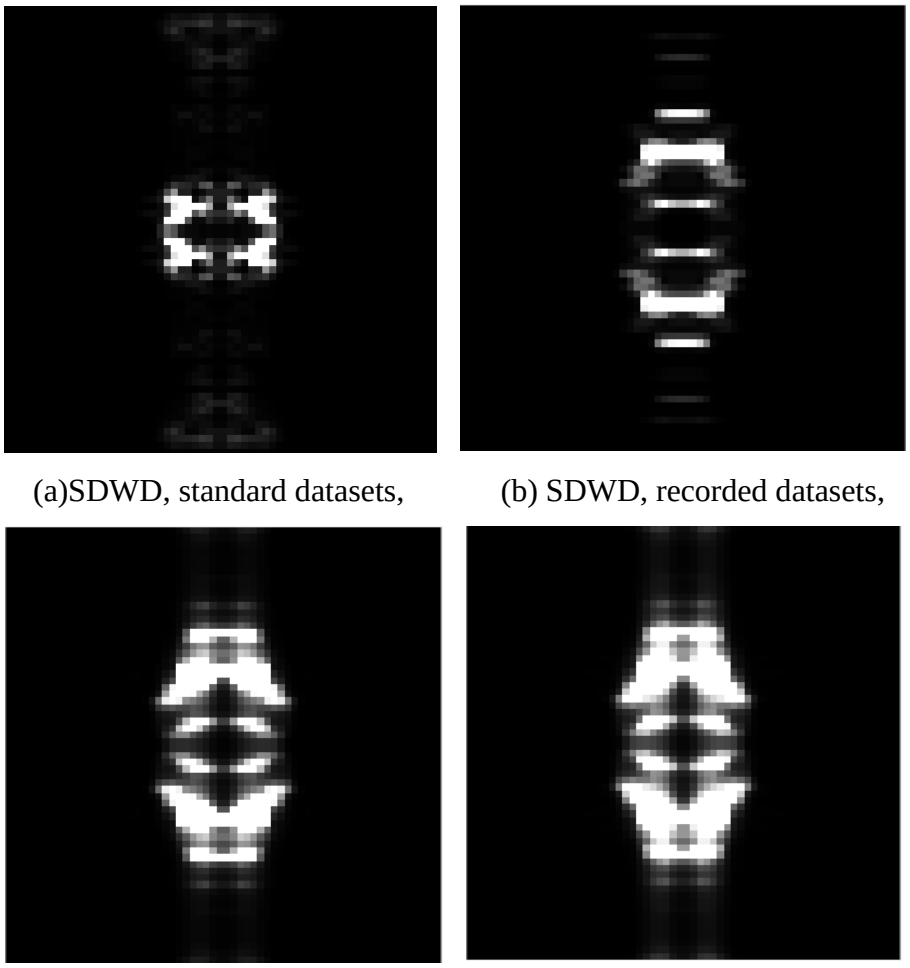
Two datasets are applied for both SDWD and BPFB, respectively. One dataset is recorded by the author. Another one is the standard dataset. The recorded dataset is the collection from two females and two males. Eight utterances were obtained for each digit per person; seven of them are used for training and the one left isused as testing data. Totally, there are three hundred and twenty utterances (320) at the sampling rate of 8 kHz. During the process of recording speech data, the ambient noise was avoided and clear pronunciation was paid attention to. The standard dataset is chosen from the big speech datasets in the website [10]. Note that each digit was kept from the same group of 4 females and 4 males. Four utterances are chosen for each digit per person. Different from the recorded dataset, the standard dataset contains much more ambient noise; unclear and various pronunciation. The sampling rate of the standard dataset is 16 kHz and it is required to be resampled to 8 kHz. Four utterances are chosen randomly as testing data for each digit and left twenty-eight ones are used for training networks.

There exist ten digits from “zero” to “nine” in each dataset. The recognition accuracy values are all calculated using the number of utterances recognized correctly over all testing utterances number (40). To show the robustness of accuracy values, all values in this section are expressed in the mean and standard deviation from running networks for ten times independently.

4.1 Two-dimensional Features

In order to apply the standard CNN, the features are constructed in the two-dimension with the frequency-axis vesus frame-axis [12]. As introduced in the section 2.4.3, two-dimensional features are composed by flipping operations to reduce the edge effect during the convolutional and pooling stages. In addition, by choosing the appropriate number of levels (7), the VAD threshold and bandwidth control value of α, the best performance can be achieved. For example, the four features for the digit “eight” are shown in Figure 4.1. For each digit, the VAD threshold

values in the recorded-SDWD, recorded-BPFB and standard-BPFB are all 0.025; and the VAD threshold in the standard-SDWD is chosen as 0.03.



(a)SDWD, standard datasets, (b) SDWD, recorded datasets,

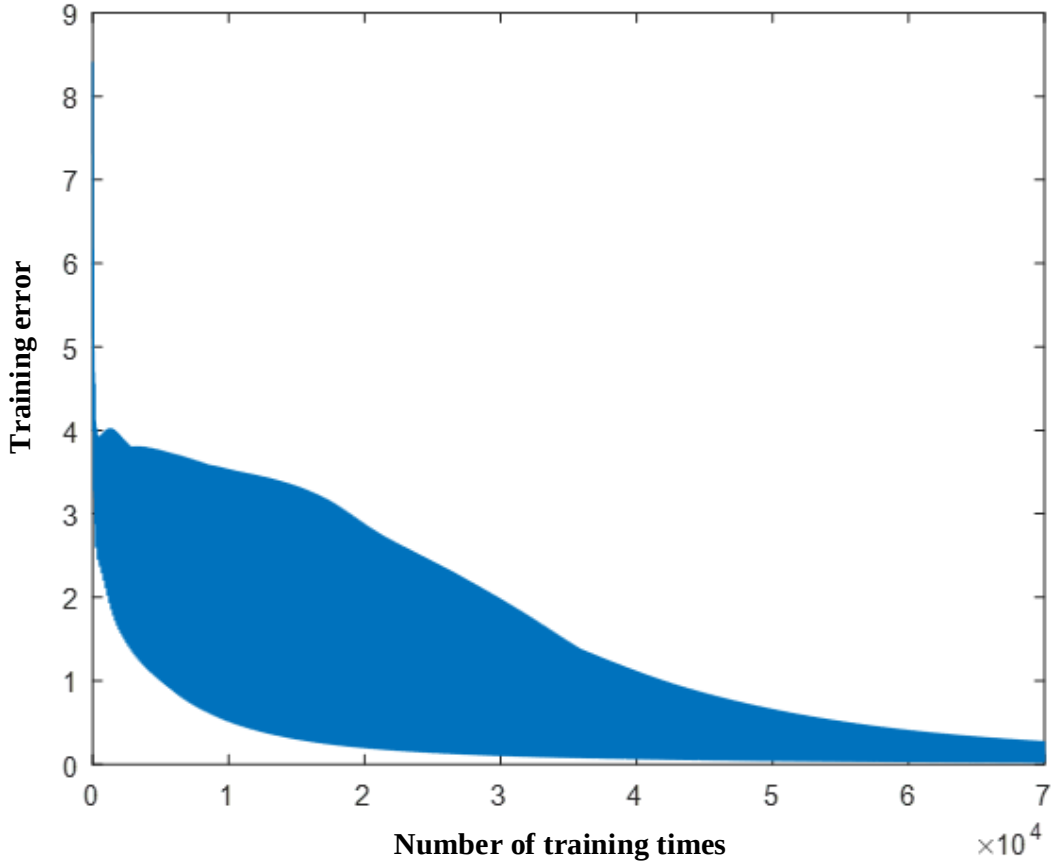
(c) BPFB, standard datasets ($\alpha = 4$), (d) BPFB, recorded datasets ($\alpha = 3$).

Figure 4.1 Flipped feature images (digit: “eight”, male).

The accuracy values are listed in the Table 4.1. The corresponding training error plots from two of the training times are shown in the Figure 4.2.

Table 4.1 Testing Accuracy of CNN (10 independent runs)

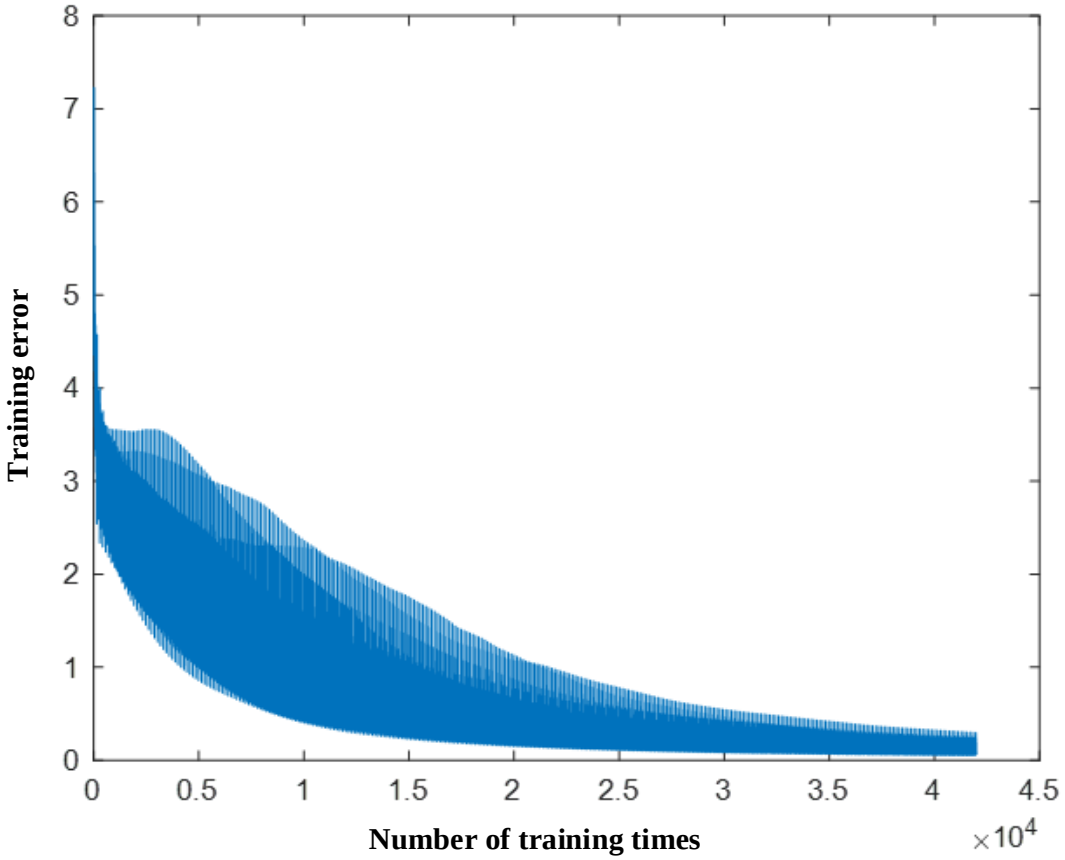
Methods / Training times	Standard datasets		Recorded datasets	
	SDWD	BPFB	SDWD	BPFB
1	90.00%	92.50%	97.50%	97.50%
2	85.00%	92.50%	97.50%	97.50%
3	85.00%	92.50%	97.50%	100.00%
4	85.00%	90.00%	97.50%	100.00%
5	85.00%	92.50%	97.50%	97.50%
6	85.00%	92.50%	97.50%	97.50%
7	85.00%	87.50%	92.50%	97.50%
8	85.00%	97.50%	97.50%	97.50%
9	87.50%	92.50%	97.50%	97.50%
10	90.00%	95.00%	97.50%	97.50%



(a) Standard-SDWD, training time 1, accuracy=90.00%,

Figure 4.2 CNN training error plots for the standard data set.

Figure 4.2 Continued.



(b) Standard-BPFB, training time 8, accuracy=97.50%.

4.2 One-dimensional Features

With the same parameters, that is, α, VAD threshold value and number of levels for decomposition, the one-dimensional features are calculated and used as inputs to the support vector machine (SVM), random forest (RF), k nearest neighbors (KNNs). Each feature has its size of 1×32 for each utterance. From Chapter 3, it is known that several parameters such as the type of kernel function, numbers of neighbors and trees, are required to adjust the performance. The achieved corresponding testing accuracy values are shown in Table 4.2. The parameters which achieve the best performance from each of networks are chosen and 10 independent runs are yielded, respectively. All accuracy values for optimal network models are shown in the Table 4.2. B. As shown in Table 4.2, the numbers of trees for standard-SDWD, standard-BPFB, recorded-SDWD and recorded-BPFB are 20, 18, 19 and 20, respectively; in Table 4.2.B, the numbers of

neighbors for KNN in these four implemtations are 3, 3, 6 and 5, respectively; again, in Table 4.2.C, the order numbers of kernel functions for the SVM are 3, 3, 3 and 2, respectively. Note that everytime the accuracy for KNN and SVM is the same with the same inputs. Thus, only one accuracy value is shown for each implemtation.

Table 4.2 Testing Accuracy of One-dimensional Features

Table A Testing Accuracy of Random Forest

Methods / Training times	Standard datasets		Recorded datasets	
	SDWD(20)	BPFB(18)	SDWD(19)	BPFB(20)
1	77.50%	87.50%	97.50%	92.50%
2	77.50%	85.00%	97.50%	90.00%
3	67.50%	87.50%	100.00%	95.00%
4	77.50%	85.00%	100.00%	92.50%
5	72.50%	85.00%	100.00%	95.00%
6	67.50%	82.50%	100.00%	90.00%
7	65.00%	87.50%	100.00%	92.50%
8	67.50%	85.00%	97.50%	95.00%
9	75.00%	87.50%	100.00%	92.50%
10	72.50%	87.50%	95.00%	97.50%

Table B Testing Accuracy Of KNN

Standard Datasets	SDWD(3)	72.50%
	BPFB(3)	82.50%
Recorded Datasets	SDWD(6)	92.50%
	BPFB(5)	92.50%

Table C Testing Accuracy Of SVM

Standard Datasets	SDWD(3)	62.50%
	BPFB(3)	67.50%
Recorded Datasets	SDWD(3)	92.50%
	BPFB(2)	92.50%

4.3 Conclusion and Future Work

The goal of this thesis work is to validate the outperformance of speech recognition for ten digits (“zero” to “nine”) using features calculated from sparse discrete wavelet decomposition (SDWD) and bandpass filter banks (BPFB) with CNN training. The CNN is implemented for speech recognition using the developed two dimentional features from the SDWD and BPFB. 2D-features are formed in terms of frame-axis and frequency-axis and further geometric transformation is applied to reduce the edge effect from the CNN convolutional layer. In order to prove the outperformance of SDWD-CNN and BPFB-CNN, the results from several common standard network algorithms using the one-dimensional features are achieved for comparisons. The standard algorithms employed include the multiclass support vector machine (SVM), random forest (RF) and k nearest neighbors (KNN). Table 4.3 shows the accuracy values of all networks employed.

Table 4.3 Conclusion of Testing Accuracy

Table A Recognition Test Accuracy Using Recorded Data Set

Method \ Accuracy		Mean accuracy	Standard deviation
CNN	SDWD	97.00%	0.0150
	BPFB	98.00%	0.0105
SVM	SDWD	92.50% (3)	0.0000
	BPFB	92.50% (2)	0.0000
RF	SDWD	98.75% (19)	1.7678
	BPFB	93.25% (20)	2.3717
KNN	SDWD	92.50% (6)	0.0000
	BPFB	92.50 (5)	0.0000

Table 4.3 Continued.

Table B Recognition Test Accuracy Using Standard Data Set

Method \ Accuracy		Mean accuracy	Standard deviation
CNN	SDWD	86.25%	2.01560
	BPFB	92.50%	0.00069
SVM	SDWD	62.50% (3)	0.00000
	BPFB	67.50% (3)	0.00000
RF	SDWD	72.00%	4.83050
	BPFB	86.00%	1.74800
KNN	SDWD	72.50% (3)	0.00000
	BPFB	82.50% (3)	0.00000

Based on Table 4.3, we can conclude that the sparse discrete wavelet decomposition and bandpass filter banks using CNN deep learning can achieve better performance than using the other standard networks. In a conclusion, the design of two-dimensional features for CNN is successful.

Through this thesis work, many contributions have been developed and are list as:

a. Developing the sparse wavelet decomposition (SDWD) based on the Mel filter banks;
b. Developing the bandpass filter banks (BPFB) based on the Mel filter banks;
c. Applying SDWD and BPFB to common standard network algorithms;
d. Applyig SDWD and BPFB to the convolutional neural network (CNN);
e. Developing the geomatric transformation to two-dimensional features for CNN.

Based on the contributions in this work, some future work could include:

a. Formulating two-dimensional features from the other different signals such as vibration signal, electroencephalogram (EEG) signal and so on, then applying the features to CNN;
b. Further studying on more possible operations of geomatirc transformation.

REFERENCES

[1] L. Rabiner, B. H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Printice-Hall, 1993.

[2] X. Huang, A. Acero, H. Hon, *Spoken Language Processing: A guide to theory, algorithm, and system development*. Prentice Hall, 2001.

[3] S. Davis, P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357-366, Agust 1980.

[4] X. He, L. Deng, W. Chou, "Discriminative learning in sequential pattern recognition—A unifying review for optimization-oriented speech recognition," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 14–36, Sep. 2008.

[5] L. Deng, X. Li, "Machine learning paradigms for speech recognition: An overview," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 21, no. 5, pp. 1060–1089, May 2013.

[6] G. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 1, pp. 30–42, Jan. 2012.

[7] Y. Bao, H. Jiang, L.-R. Dai, and C. Liu, "Incoherent training of deepneural networks to de-correlate bottleneck features for speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, pp. 6980–6984, May 2013.

[8] K. F. Lee and H. W. Hon, "Speaker-independent phone recognition using hidden Markov models," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 37, no. 11, pp. 1641–1648, Nov. 1989.

[9] J. Dai, V. Vijayarajan, X. Peng, L. Tan, J. Jiang, "Speech recognition using sparse wavelet decomposition features," *2018 IEEE International Conference on Electro/Information Technology*, pp. 812-816, May 2018.

[10] https://www.tensorflow.org/tutorials/sequences/audio_recognition #preparation

[11] L. Tan, J. Jiang, *Digital Signal Processing: Fundamentals and Applications.* Third Edition, Elsevier/Academic Press, 2018.

[12] J.Dai, Y. Zhang, J. Hou, X. Wang, L. Tan, J. Jiang, “Speech wavelet decomposition and filter banks with CNN deep learning for speech recognition,” *submitted to 2019 IEEE International Conference on Electro/Information Technology*, May 2019 .

[13] https://en.wikipedia.org/wiki/Bilinear_interpolation

[14] Steve.R.Gunn, *Support Vector Machine for Classification and Regression*. *Technical Report*, 10 May 1998.

[15] https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#intro

[16] https://en.wikipedia.org/wiki/Decision_tree_learning

[17] https://en.wikipedia.org/wiki/Random_forest#Bagging

[18] Altman, N. S. *(1992). "An introduction to kernel and nearest-neighbor nonparametric regression". The American Statistician. 46 (3):175-185.* doi:10.1080/000311305.1992. 10475879. hdl: 1813/31637.

[19] https://people.revoledu.com/kardi/tutorial/KNN/

[20] https://en.wikipedia.org/wiki/Euclidean_distance

[21] A. Sharma, A. Suryawanshi, “A novel method for detecting spam email using KNN classification with spearman correlation as distance measure,” *International Journal of Computer Applications (0975-8887)*, vol.136, no.6, Feb. 2016.

[22] Collobert, Ronan; Weston, Jason (2008-01-01). “A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning,” *Proceedings of the 25th International Conference on Machine Learning*, pp. 160–167, January 2008.

[23] P. Ongsulee, “Artificial intelligence, machine learning and deep learning,” *2017 International Conference on ICT and Knowlegde Engineering (ICT&KE)*, Nov. 2017.

[24] "CS231n Convolutional Neural Networks for Visual Recognition". cs231n.github.io. Retrieved 2018-12-13.

PUBLICATIONS

Conference Papers:

V.Vijayarajan, J.Dai, L.Tan, J.Jiang, “Channel Sparsity-Aware Diagonal Structure Volterra Filters for Nonlinear Acoustic Echo Cancellation” 2018 IEEE INTERNATIONAL CONFERENCE ON ELECTRO INFORMATION TECHNOLOGY. PP, 420-423, Oakland University, Rochester, Michigan, May 2018

J.Dai, V.Vijayarajan, X.Peng, L.Tan, J.Jiang, “Speech recognition using sparse discrete wavelet decomposition feature extraction” 2018 IEEE INTERNATIONAL CONFERENCE ON ELECTRO INFORMATION TECHNOLOGY, pp. 812-816, Oakland University, Rochester, Michigan, May 2018.