SPEAKER AND POSTURE CLASSIFICATION USING INSTANTANEOUS ACOUSTIC FEATURES OF BREATH SIGNALS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ATIL İLERİALKAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MULTIMEDIA INFORMATICS

NOVEMBER 2019

Approval of the thesis:

SPEAKER AND POSTURE CLASSIFICATION USING INSTANTANEOUS ACOUSTIC FEATURES OF BREATH SIGNALS

submitted by **ATIL İLERİALKAN** in partial fulfillment of the requirements for the degree of **Master of Science in Modelling and Simulation Department, Middle East Technical University** by,

Prof. Dr. Deniz Zeyrek Bozşahin
Dean, Graduate School of **Informatics** _____

Assist. Prof. Dr. Elif Sürer
Head of Department, **Modelling and Simulation, METU** _____

Assoc. Prof. Dr. Hüseyin Hacıhabiboğlu
Supervisor, **Modelling and Simulation, METU** _____

Prof. Dr. Alptekin Temizel
Co-supervisor, **Modelling and Simulation, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Banu Günel Kılıç
Information Systems Department, METU _____

Assoc. Prof. Dr. Hüseyin Hacıhabiboğlu
Modelling and Simulation Department, METU _____

Prof. Dr. Alptekin Temizel
Modelling and Simulation Department, METU _____

Assist. Prof. Dr. Tolga İnan
Electrical and Electronics Engineering, Çankaya University _____

Assist. Prof. Dr. Hacer Yalım Keleş
Computer Engineering Department, Ankara University _____

Date: 27.11.2019

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Atıl İLERİALKAN

Signature :

ABSTRACT

SPEAKER AND POSTURE CLASSIFICATION USING INSTANTANEOUS ACOUSTIC FEATURES OF BREATH SIGNALS

İLERİALKAN, Atıl

M.S., Department of Modelling and Simulation

Supervisor: Assoc. Prof. Dr. Hüseyin Hacıhabiboğlu

Co-Supervisor : Prof. Dr. Alptekin Temizel

November 2019, 65 pages

Acoustic features extracted from speech are widely used for problems such as biometric speaker identification or first-person activity detection. However, use of speech data raises concerns about privacy due to the explicit availability of the speech content. In this thesis, we propose a method for speech and posture classification using intra-speech breathing sounds. The acoustical instantaneous side information was extracted from breath instances using the Hilbert-Huang transform. Instantaneous frequency, magnitude, and phase features were extracted using intrinsic mode functions, and different combinations of these were fed into a CNN-RNN network for classification. We also created a publicly available breath dataset, BreathBase, for both our experiments in the thesis and future work. BreathBase contains more than 5000 breath instances detected on the recordings of 20 participants reading pre-prepared random pseudo texts in 5 different postures with 4 different microphones. Using side information acquired from breath sections of speech, 87% speaker classification and 98% posture classification accuracy is obtained among 20 speakers with this method. The proposed method outperformed various other methods such as support vector machines, long-short term memory and combination of k-nearest neighbor and dynamic time warping techniques.

Keywords: speaker recognition, posture recognition, hilbert huang transform, instantaneous frequency

ÖZ

NEFES SİNYALLERİNİN ANLIK AKUSTİK ÖZELLİKLERİNİ KULLANARAK KONUŞMACI VE DURUŞ SINIFLANDIRMASI

İLERİALKAN, Atıl
Yüksek Lisans, Modelleme ve Simülasyon Bölümü
Tez Yöneticisi: Doç. Dr. Hüseyin Hacıhabiboğlu
Ortak Tez Yöneticisi : Prof. Dr. Alptekin Temizel

Kasım 2019 , 65 sayfa

Konuşmadan çıkarılan akustik özellikler, biyometrik konuşmacı tanımlama veya birinci şahıs eylemlerinin kestirimi gibi problemlerde yaygın olarak kullanılır. Ancak, konuşma verilerinin kullanımı, konuşma içeriğinin açık bir şekilde kullanılabilir olması nedeniyle gizlilik konusundaki endişeleri artırmaktadır. Bu tezde konuşma aralarındaki nefes verilerini kullanarak konuşma ve vücut pozisyonu sınıflandırması için bir yöntem öneriyoruz. Bu yöntemde akustik anlık yan bilgi, Hilbert-Huang dönüşümü kullanılarak nefes örneklerinden çıkarılır. Anlık frekans, büyüklük ve faz özellikleri, içsel kip işlevleri kullanılarak çıkarılır ve bunların farklı kombinasyonları, sınıflandırma için CNN-RNN ağına beslenir. Ayrıca, hem bu tezdeki deneylerimiz hem de gelecekteki çalışmalarımız için genel erişime açık bir nefes veri seti, BreathBase'i oluşturduk. BreathBase, önceden hazırlanmış rastgele sözler içeren metinleri 4 farklı mikrofonla 5 farklı vücut pozisyonunda okuyan 20 katılımcının kayıtlarında tespit edilen 5000'den fazla nefes örneği içermektedir. Konuşmanın nefes bölümlerinden elde edilen yan bilgileri kullanarak, bu yöntemle 20 konuşmacı arasında % 87 konuşmacı sınıflandırma ve % 98 duruş sınıflandırma doğruluğu elde edilmiştir. Önerilen ağ ayrıca SVM, LSTM ve kNN-DTW tekniklerinin birleştirilmesi gibi diğer yöntemlerden daha iyi performans göstermiştir.

Anahtar Kelimeler: konuşmacı tanıma, duruş tanıma, hilbert huang dönüşümü, anlık frekans

This thesis is dedicated to my family, my dear girlfriend, my friends, teammates and the people who care about me.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Assoc. Prof. Dr. Hüseyin Hacıhabiboğlu and my co-advisor Prof. Dr. Alptekin Temizel for continuous support during my thesis and my studies for the master's degree. Their knowledge and guidance helped me through during this journey.

Secondly, I would like to thank my family and my girlfriend for being fully supportive of my academic career since the beginning until the last moment. Without their support, I wouldn't be able to accomplish this task by any chance. A special thanks to my dearest, also for being the joy of life for better, for worse.

Thanks to my cats, Java and Pixel, who helped me by providing all the love they have and keeping me high-spirited. Also, I would like to thank Bilgin Aksoy and all my friends in Hacettepe Reddeers american football team for all their efforts and help during this challenging passage of my life.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

SPARG	Spatial Audio Research Group
MFCC	Mel-Frequency Cepstral Coefficient
IMF	Intrinsic Mode Function
EMD	Empirical Mode Decomposition
HHT	Hilbert Huang Transform
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long-Short Term Memory
GRU	Gated Recurrent Unit
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
RMSprop	Root Mean Square Prop(agation)
Adam	Adaptive Moment Estimation
kNN	k-Nearest Neighbors
DTW	Dynamic Time Warping

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

Breathing is a central part of human speech and while it frequently happens during speech production, it is often ignored as it does not contain any distinct perceptual information. However, it can easily be used as an input into biometric verification systems [1] as it is impossible to imitate because it depends on the speaker's physiology (e.g. lung volume, larynx, oral and nasal cavities) that varies strongly from person to person. Therefore, it can be used as a control and verification mechanism against speech-based biometric attacks and impersonations [2]. Due to the privacy and data protection laws and regulations, the storage and use of spoken content may not always be possible. On the other hand, breathing can be processed without using any speech content and converted into useful side-channel information by way of acoustic signal processing. It is a very universal and valuable tool since the information to be extracted from the breath is independent of the spoken content and the language used. In the case of people with disabilities and during emergencies, any information that can be obtained from breathing sounds could be of great importance. In this work, we explore the use of the information gathered from breath signals.

1.2 Proposed Methods and Models

We propose methods of feature embedding using Hilbert-Huang Transform and network architectures consisting of convolutional and recurrent neural network layers to solve a classification problem using this time-series, multi-class data.

The proposed approach is based on the observation that breathing signals are not stationary, meaning that Fourier-based methods —at least theoretically— are not appropriate for analyzing breathing sounds. For that reason, we proposed a method using features extracted from breathing sounds using the Hilbert-Huang transform. Namely, we use time-varying instantaneous frequencies, instantaneous magnitudes, and instantaneous phases as features that are fed to an RNN network which is known to be an effective classifier for time series data. The classifier is trained using data labeled for different postures and speakers. The results indicate that the proposed approach is feasible in classifying both postures and speakers.

1.3 Contributions and Novelties

The main contributions of this thesis are as follows:

- Development of a new and publicly available dataset consisting of human breaths, BreathBase.
- Use of the instantaneous features extracted from breath instances for speaker and posture classification.
- A speaker and posture classification model using instantaneous acoustical features.

1.4 The Outline of the Thesis

This thesis is structured as follows. In the second chapter, current studies in the field and theoretical information about the extracted features are presented. In the third chapter, more practical information is given with representations and reasons for the choices of used methods. The fourth chapter provides information on the collected data, phases of collection and collecting environment. Experimental evaluation and results are provided in the fifth chapter. A chapter consisting of final thoughts and conclusive inferences ends this thesis.

CHAPTER 2

BACKGROUND

2.1 Current Literature on Acoustical Analysis of Human Breath

Analysis of human breath is a prevalent topic in areas such as health sciences and engineering. Breathing sounds can be used to reveal a variety of different information about a speaker because the process of breathing depends on specific physiological condition of the speaker [3]. Diagnosis of respiratory diseases and biometric verification are some sample applications using human breathing sounds. For the intentions of this thesis, we will mainly focus on the studies in the field of engineering. In the area, mostly the detection and exact demarcation of human breath and its phases have been studied. Among these, some studies use mel-frequency [4, 5, 6, 7] or i-vector [8] based approaches for feature extraction, while others use raw breath recordings [2, 9, 10]. Automatic classification of breath types and phases is another area of study [11, 12, 13]. As for speaker recognition, some studies work with breathing alone, while others include all non-speech sounds in association with breath signals [8, 2, 14, 15, 16, 17, 1]. Some of them have focused on the detection and removal of breath signals and other non-speech sounds in order to exclude them for a better speech recognition [18, 19]. Some specific studies only deal with nasal breathing or inhalation, while the majority do not [12, 20]. The relationship between breathing statistics and human emotions is also among the research areas [21, 22]. In the data sets subject to the studies, various participants or recordings from theater actors to vocal artists, songs to television and telephone records were used [5, 6, 7, 23]. In order to collect these recordings, many different devices were used together with or without microphones, including different near/far-field sensors, imaging devices and respirators, especially used in the health field [24]. Most of the engineering studies in the field of health were carried out for the detection of snoring and respiratory diseases and anomalies [10, 13, 25, 26].

2.2 Stationarity of Breath

On a long-term basis, speech signals are not stationary. This being said, when the windows of 20-30 milliseconds of sizes were analyzed over time, it is assumed to provide approximately stationary characteristics because of their slowly-varying nature. In fact, due to this behavior, speech signals are often referred to as quasi-stationary or short-time stationary signals [27, 28, 29]. However, non-stationary parts of the speech such as various accents, emphasis and different pronunciations of monophthongs and

diphthongs challenges this stationarity even inside of the given analysis window [30]. Besides, methods admitting this stationarity (i.e., mel-frequency cepstral coefficients) cannot accurately detect localized events like intra-speech breathings [31].

Breath signals, on the other hand, are not stationary even when examined in short periods because of the changing levels and/or direction of the airflow, changing volume of lungs and changing turbulence around the mouth while breathing[32, 33]. It also shows variance depending on age, mass, current state and history of pathologic and physiologic state [34, 35, 36]. In order to assess the stationarity of breathing sounds, we ran the Augmented Dickey-Fuller (ADF) test on breath samples. ADF test is a statistical hypothesis test. Its null hypothesis is that a unit root is present in a time series data, meaning non-stationarity. So, more negative ADF statistic means more stationary data. Test also yields p-value and critical ADF values for comparison. The below table summarizes the number of stationary and non-stationary samples according to different probable critical values. It also shows the percentage of non-stationary samples to all samples in the dataset.

Table 2.1: Classification Results of ADF Test for BreathBase

Critical p-value = 0.01			**Critical p-value = 0.05**			**Critical p-value = 0.10**		
St.	**Non-St.**	**%**	**St.**	**Non-St.**	**%**	**St.**	**Non-St.**	**%**
783	4287	84.56%	999	4071	80.3%	1177	3893	76.79%
Critical Adf-Stat = 0.01			**Critical Adf-Stat = 0.05**			**Critical Adf-Stat = 0.10**		
St.	**Non-St.**	**%**	**St.**	**Non-St.**	**%**	**St.**	**Non-St.**	**%**
232	4838	95.43%	419	4651	91.74%	802	4268	84.19%

After reviewing the results in Table 2.1, ADF test revealed that stationarity does not hold for most of the tested samples. Having established the non-stationarity of breathing sounds, it is not meaningful to use features extracted using Fourier series-based methods. For example, it is practically possible to analyze breath signals using MFC Coefficients, but in theory, it does not provide detailed information that can be used to analyze detailed psychophysical status. Therefore, we used the Hilbert-Huang transform (HHT) method; which does not assume stationary or linear processes and provides instantaneous frequency, phase and amplitude data, can be used in the analysis of such signals. In this thesis, we extracted these instantaneous features from breath signals and classified them using several machine learning methods.

2.3 Hilbert-Huang Transformation

The Hilbert-Huang transformation (HHT) is useful for performing time-frequency analysis on signals and data from nonlinear and non-stationary processes, which most of the natural processes are. Other than sound processing [37, 38, 39, 40, 41, 42, 43], HHT has various applications in engineering, biomedical, financial and geophysical

fields [44]. It aims to overcome one of the most important constraints of Fourier-based methods (such as short-time Fourier transform), the (short-time) stationarity. Instantaneous frequency (including instantaneous amplitude and phase) extracted from the data allows an analysis and the characterization of the underlying process.

The method is based on decomposing signal into intrinsic mode functions (IMF), which are recursively obtained from the signal itself, rather than the complex exponential functions used in the Fourier transform. HHT consists of two stages; empirical mode decomposition and Hilbert transform [45].

2.3.1 Empirical Mode Decomposition

One major problem for the instantaneous frequency to be practical was its use on functions that consist of multiple other functions, so-called "modes". Empirical mode decomposition resolved this problem by expressing a non-stationary, multicomponent signal as a linear combination of multiple intrinsic mode functions (IMF) with different amplitudes and from which instantaneous frequencies can be calculated.

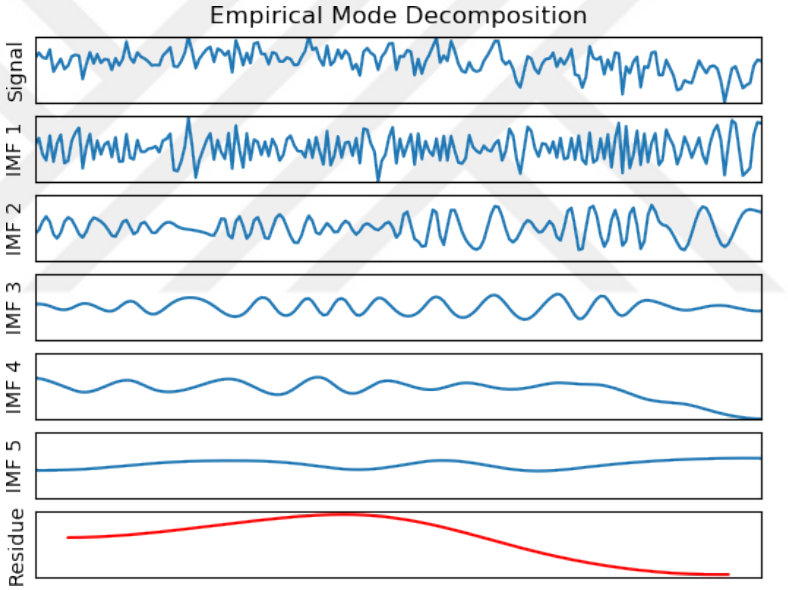


Figure 2.1: Decomposition of a section of a breath sample

An $x(t)$ signal with time support between $[0, T]$ which is non-stationary, consisting of some unknown p number of IMFs, can be expressed as follows:

$$x(t) = \sum_{k=1}^{p} c_k(t) + r(t) \tag{2.3.1}$$

where $c_k(t)$ is the kth IMF and $r(t)$ is the residual signal. These IMFs are obtained recursively by a sifting process. This process is carried out algorithmically as follows until a certain stop condition is met.

- First, all local extremes in the signal are found.

- All local maximum and minimum values are combined to form upper and lower envelopes, respectively, using cubic splines.

- Accordingly, the average of the envelopes (m_1) is calculated and subtracted from the original signal ($x(t)$): $h_1 = x(t) - m_1$

- After that, h_1 is checked for criteria for being an IMF (defined in 2.3.1.1). If not satisfying these, h_1 treated as proto-imf and this process is repeated until it does, using itself as data: $h_{11} = h_1 - m_{11}$

- After sifting is repeated for n iterations, h_{1n} becomes the first imf (c_1) and subtracted from the original signal: $r_1 = x(t) - c_1$

- The remaining residue signal (r_1) is handled as data and all steps are repeated until there is no IMF remaining in the residue signal or a stopping condition.

After each cycle of decomposition, the information remaining in the residue signal decreases, so each IMF carries less information than its predecessor. Therefore, the number of already decomposed IMFs or the stationarity of the remaining residue signal may be defined as a stopping criterion for the number of IMFs (n_{imf}) extracted from the signal[46].

2.3.1.1 Intrinsic Mode Functions

Intrinsic mode functions are monocomponent functions which satisfy the following conditions:

- number of zero crossings differ from the number of extrema with a maximum value of one and

- mean value of the envelope is close to zero.

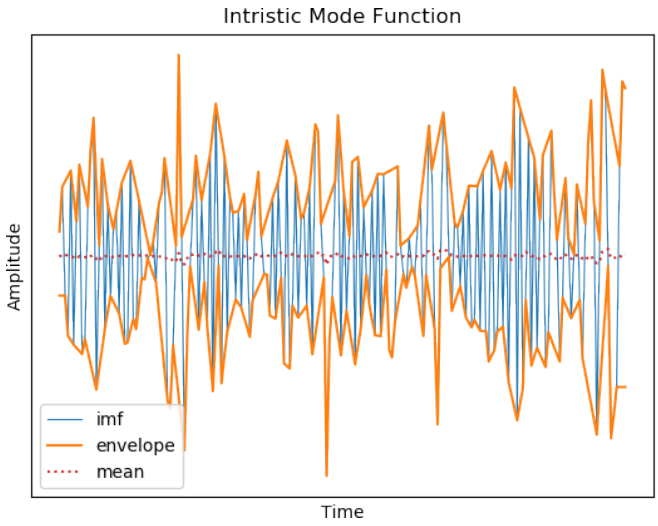
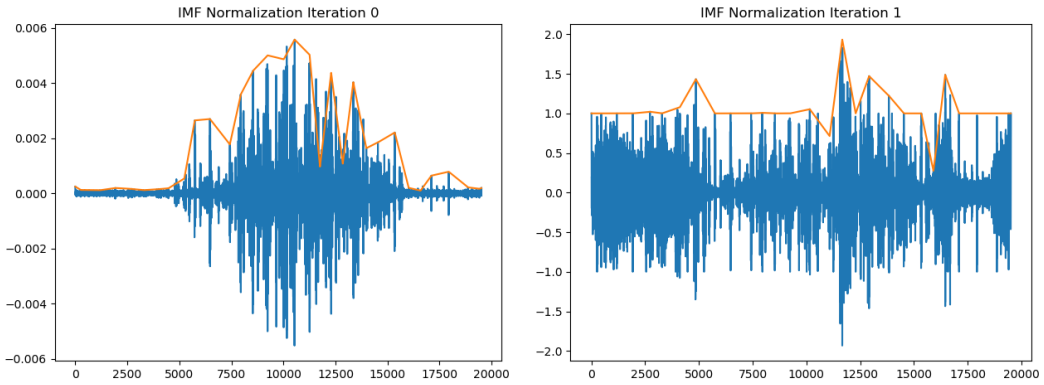


Figure 2.2: Intrinsic Mode Function of a section of a breath sample

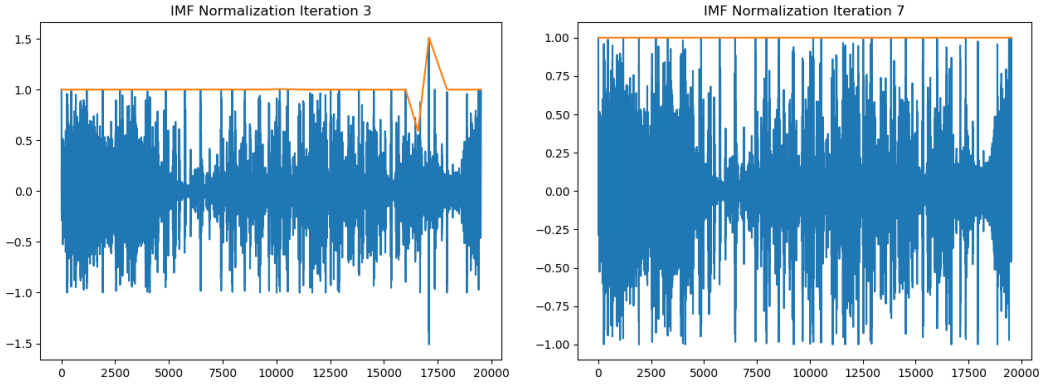
2.3.1.2 IMF Normalization

Instantaneous frequencies(or frequency), by definition, should not have negative values. However, for IMF signals whose amplitude-modulated envelope is changing directions frequently (not slowly variant as opposed to explanations in section 2.3.2), negative values are inevitable [47]. For more accurate calculations of instantaneous frequencies, these IMF signals should be normalized first. This process is carried out with, first constructing an envelope from extremes of the absolute value of the IMF is obtained. Then, the IMF is normalized by dividing it with this envelope. This process is repeated until the IMF's envelope resembles a straight line with an identical value at the desired precision prior to Hilbert Transform. These normalizations solve most of the problems, but in rare conditions, they can still occur [48].



(a) Original Signal

(b) First Iteration



(c) Third Iteration

(d) Last Iteration

Figure 2.3: Steps of an IMF Normalization Process

2.3.2 Hilbert Transform

For extracting instantaneous frequency, Hilbert Transform is applied to any real-valued signal $x(t)$ to get an $analytical$ signal with the complex conjugate $y(t)$. The analytic pair $z(t)$ then can be defined as:

$$z(t) = x(t) + iy(t) = a(t)e^{i\theta(t)} \tag{2.3.2}$$

where $a(t)$ being instantaneous amplitude as:

$$a(t) = \sqrt{x(t)^2 + y(t)^2} \tag{2.3.3}$$

and $\theta(t)$ being phase function as:

$$\theta(t) = \arctan(\frac{y(t)}{x(t)}). \tag{2.3.4}$$

From there, the instantaneous frequency can be defined as:

$$\omega = \frac{\partial \theta}{\partial t}. \tag{2.3.5}$$

Please note that these equations hold for any slowly varying function according to the classical wave theory [49].

For each IMF signal, firstly Hilbert transform is applied to construct imaginary parts of real-valued signals and obtain an analytical signal. The original signal, as the actual part of the analytical signal, can be expressed as follows:

$$x(t) = \Re\Big[\sum_{k=1}^{p} a_k e^{i\omega_k(t)t}\Big] \tag{2.3.6}$$

Here, a_k and ω_k are constant, amplitude and frequency modulations are also separated from each other. This frequency-time distribution of the signal amplitude is called the Hilbert spectrum. From this formula, instantaneous energy ($\|a_k\|^2$) and instantaneous frequency ($\frac{\partial \omega_k}{\partial t}$) can be calculated. If we calculate the instantaneous frequency, first, the phase of the analytical signal is calculated and unwrapped. Then, the derivative of this phase is taken and the instantaneous frequency(in Hertz) is found with the following formula:

$$f_{instant}(t) = \frac{1}{2\pi}\frac{\partial \theta}{\partial t} \times F_s \tag{2.3.7}$$

where F_s is the sampling frequency of the discrete time signal.

Below is a graphical comparison of the instantaneous frequency and the short-time Fourier transform (STFT) generated from an amplitude modulated sinusoidal chirp signal of which the frequency increases from 20 Hz to 100 Hz. It may be observed that instantaneous frequency extracted using HHT is able to identify changes in the frequency exactly, while STFT cannot accurately identify them, but approximates.

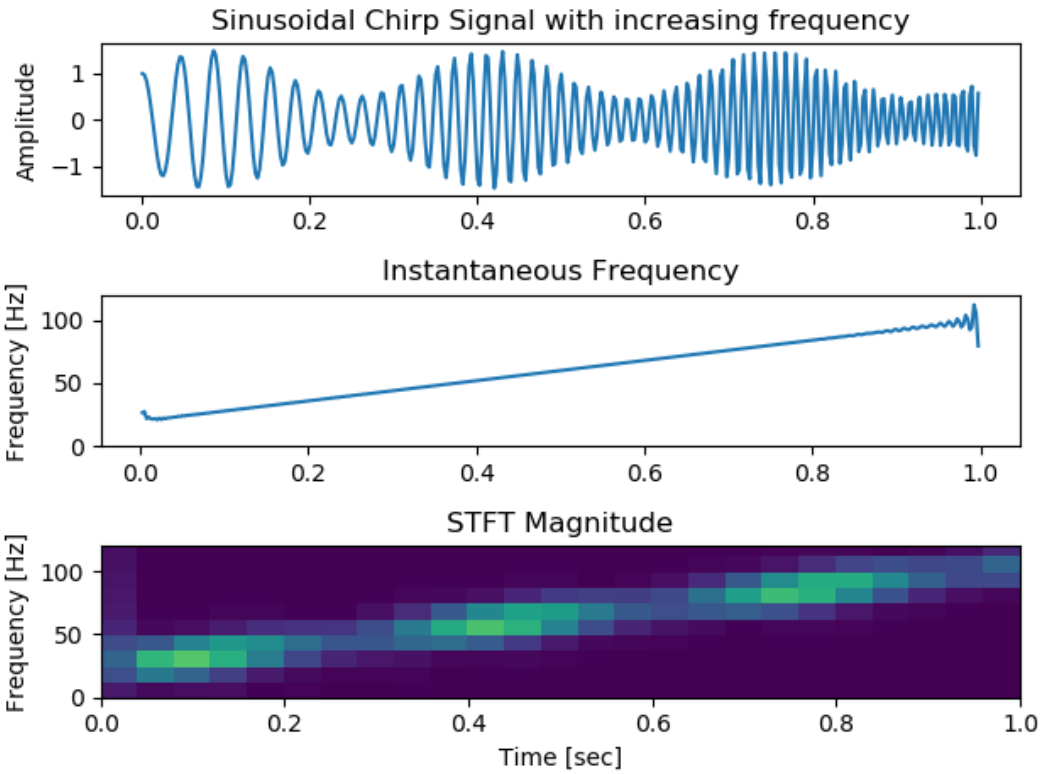


Figure 2.4: HHT vs STFT

CHAPTER 3

DATA COLLECTION

3.1 Participants

Breath data used for the experiments were collected in the Spatial Audio Research Group (SPARG) Laboratory in Modelling and Simulation Center in METU. A total of 20 volunteers (will be referred to as participants hereafter) who are active licensed football players participated in the data collection study. Participants were selected among professional athletes since their good medical condition is endorsed by medical professionals. A copy of medical reports was presented by each volunteer prior to recording.

Gender and age diversity was also taken into consideration for the validity and integrity of the research findings. An equal number of men and women (10 from each) ranging from 20 to 30 years old attended the study.

3.2 Text

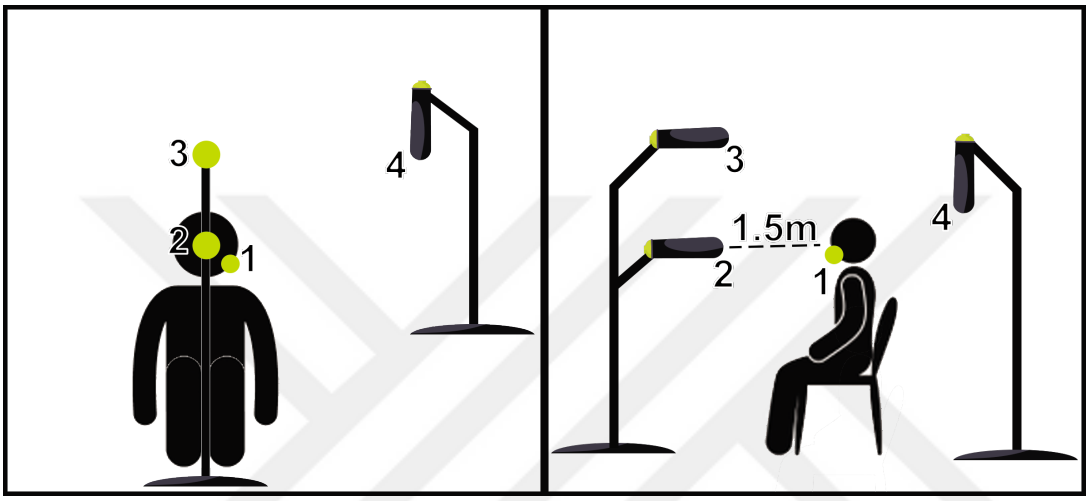
Each participant was requested to read one and a half pages long texts from "The Sorrows of Young Werther" of Johann Wolfgang von Goethe and "The Metamorphosis" of Franz Kafka. The sequence of original texts was randomized and translated to Turkish with Google Translator to make them unintelligible with the purpose of not giving any emotional cues or orientations to the participants. In order to avoid the occurrence of pauses like silences, emphasis and filled pauses (like 'umm' voices) while reading, the entire text was decapitalized and all punctuations were removed.

3.3 Recording Postures

Each participant was handed a randomized, slightly different text to read in 5 different postures (sitting, low sitting, standing, hands behind head, lying). Details of these postures are given in the remainder of this section.

3.3.1 Sitting Posture

This position gives a default sitting posture to the speaker with both hands holding the text and facing forward as possible. Lower one of the frontal microphones is aligned to be at the same level as the speaker's head with a distance of 1.5 meters.

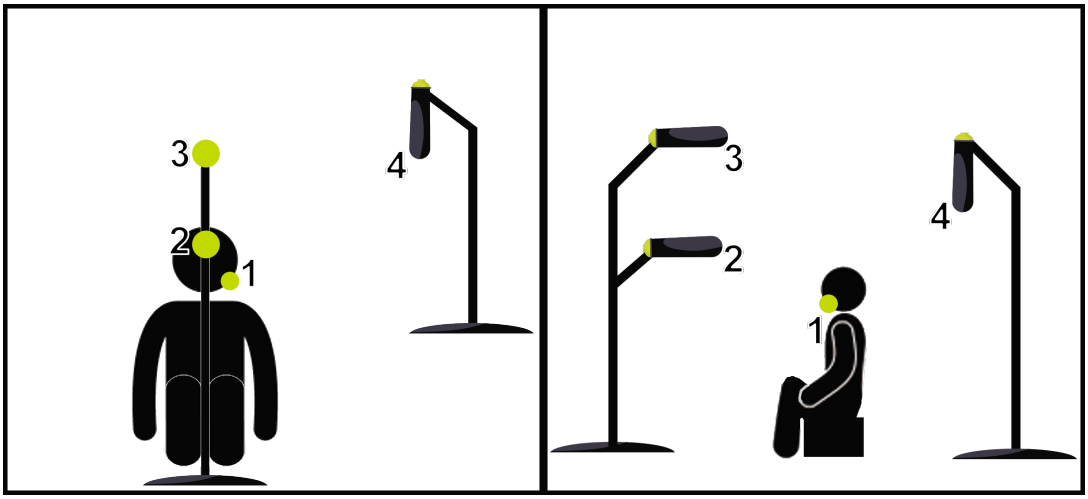


(a) Front view (b) Side view

Figure 3.1: Sitting Position

3.3.2 Low Sitting Posture

This position gives a lower sitting posture to the speaker with both hands holding the text, legs and knees pressuring diaphragm, facing forward as possible. The lower frontal microphone is aligned to be at a distance of 1.5 meters from the speaker's head, horizontally.

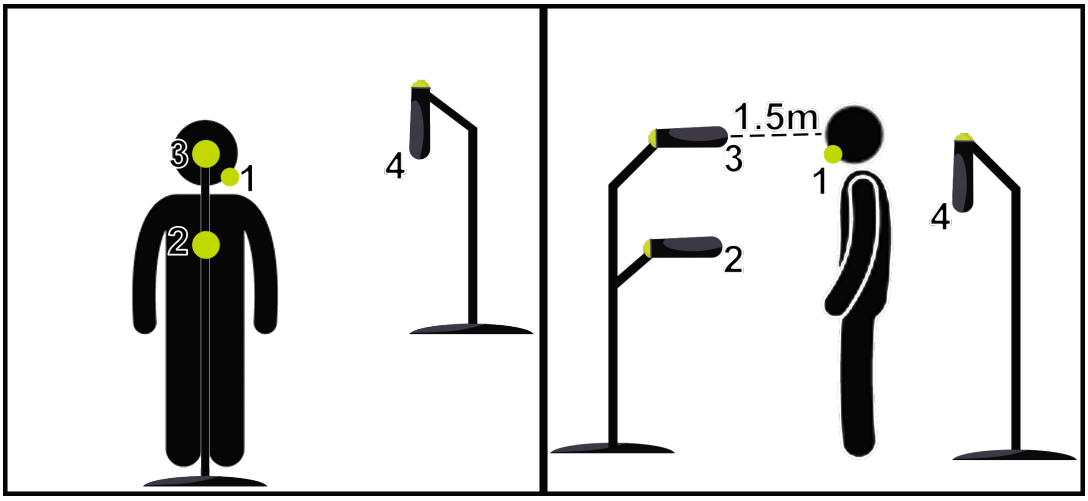


(a) Front view (b) Sideview

Figure 3.2: Low Sitting Position

3.3.3 Standing Posture

This position gives default standing posture to the speaker with both hands holding the text, facing forward as possible. The higher frontal microphone is aligned to be at the same level as the speaker's head with a distance of 1.5 meters.

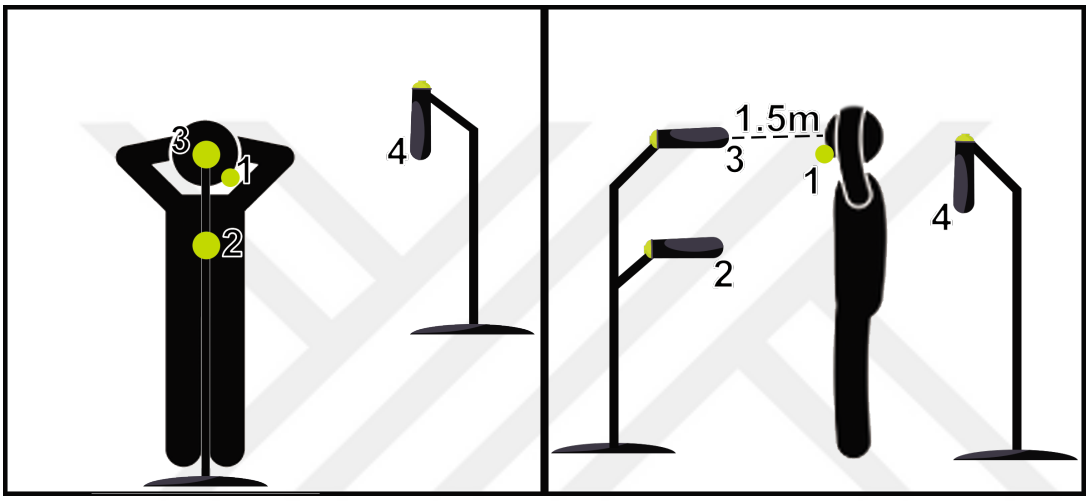


(a) Front view (b) Sideview

Figure 3.3: Standing Position

3.3.4 Hands Behind Head Posture

This position gives an alternative standing posture to the speaker with hands behind the head, providing as much volume to the lungs as possible, also facing forward as possible. Text is standing on a table in front of the speaker, 1.3 meters above the floor, easily readable. Higher one of the frontal microphones is aligned to be at the same level as the speaker's head with a distance of 1.5 meters.

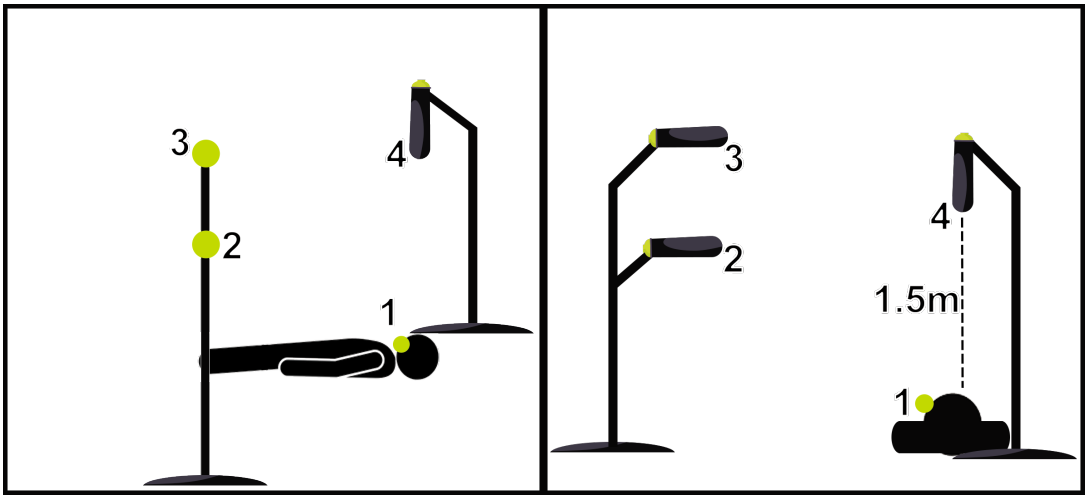


(a) Front view (b) Sideview

Figure 3.4: Hands Behind Head Position

3.3.5 Lying Posture

This flat position with chest up alignment gives an alternative posture to the other positions with regards to the direction of the gravitational force. Speaker lies while holding the text with both hands, facing upwards as possible. Another microphone is aligned to be 1.5 meters above the face of the speaker.



(a) Front view (b) Sideview

Figure 3.5: Lying Position

3.4 Studio

Recordings were collected at the METU SPARG Lab which is a specially designed acoustic booth with very low reverberation ($T_{60} = 80ms$). The studio room has no parallel walls, including the floor and the ceiling in order to eliminate standing waves. As a result of this specific design and sound insulation materials used for the construction of the studio, it has a maximum background noise level of 40 dB (SPL).

3.5 Recording Setup

3.5.1 Microphones

Near- and far-field microphones were used for the recordings. Far-field microphones were placed considering having equal distanced microphones from the speaker's head for most of the recording postures and kept stable for each recording. Far-field microphones used for the recordings are three of Rode M5 cardioid condenser microphones. One pair was used in front of the speaker with 1.3 and 1.7 meters above ground, respectively, and at a 1.5 meters horizontal distance from the speaker's head during sitting and standing positions. The third one was placed 1.7 meters above the ground, facing down, approximately 1.5 meters away from lying speaker's face. The horizontal distance of this microphone to the standing and sitting positions were also set to 1.5 meters.

Near-field microphone was a DPA 4060 lavalier microphone. It was place placed next to the speaker's mouth.

3.5.2 Audio Interface

MOTU 896mk3 Hybrid Audio Interface was used for translating 4 channel, analog signal to 48kHz sampled digital signal with Audacity 2.1.2.0 software. Microphones were assigned to channels as follows:

- Channel 1: Near-field (placed near the mouth)
- Channel 2: Front-Higher
- Channel 3: Front-Lower
- Channel 4: Behind

3.5.3 Recording Process

First, each participant was familiarized with the laboratory and the recording studio. All the positions were demonstrated on-site and following instructions were given:

- The text you will read is purposefully devoid of meaning. Do not pay attention to the content while reading. Read as natural and neutral as possible.
- Read your name and position before starting a text in a new position.
- Try to face forward in every posture.

After the participant was informed about the process and instructions, the first text was handed and recording started once the door of the recording room was closed. After the participant completes the reading, the recording is finished and the instructor enters the recording room to set it up for the next position. Position list is given in section 3.3. When all of the positions are complete, recordings are trimmed to have only readings and they are ready for breath extraction.

3.5.4 Dataset

For splitting breath portions from voices, a set of basic methods were applied. First, the recording was divided into sections with 1-millisecond length and energies of these sections were calculated. It is known that the level of a normal human conversation is around 60-70 dB SPL. When the energy of a section is dropped under 40dB, that section was marked as the beginning of a breath. Any next section which has more than 25dB energy was marked as the end of the breath. It is also known that breath signals have edges on both ends, just before and after the breath event [7]. Thus, every marked breath candidate was trimmed from both ends using minimum energy points of both halves. Since intra-speech breathing takes around 0.2 to 1.0 seconds [6], final duration thresholding was applied and final breath sections were acquired.

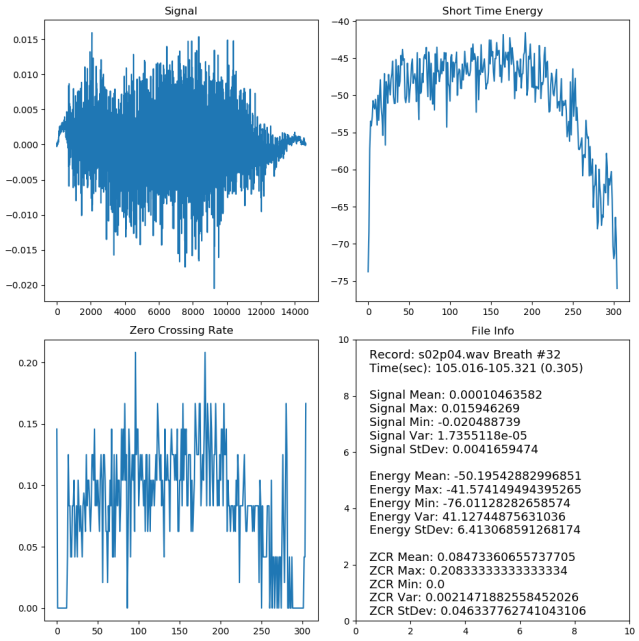


Figure 3.6: A sample breath candidate plot

Zero crossing rate parameter was also investigated for better demarcation, but significant improvement over level-based segmentation was not observed. Breath candidates were compared with human markings and all breath events were found with a false positive rate of 0.2479. When these false positives were eliminated, a total of 5070 breath instances were selected (out of 6742 candidate) for 20 participants and 5 postures for each participant.

3.5.5 Post-Processing

After recordings were completed, we noticed a background noise generated in lower frequencies of the recordings of far-field microphones. This behavior is due to the acoustic characteristics of the diaphragm of the condenser microphones [50]. To remove the effects of this noise, we applied a simple high-pass (roll-off) filter with a cutoff frequency of 70Hz and a filter size of 4097.

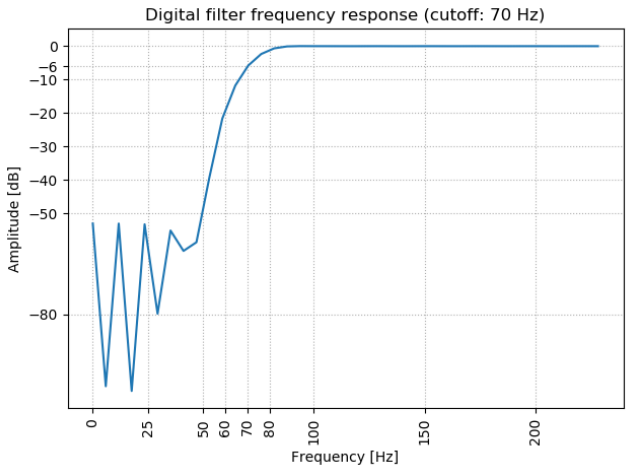


Figure 3.7: Magnitude response of the high-pass filter used for removal of the low-frequency noise from the recordings in the database

The effects of the filter on a breath instance can be seen on each channel in the following figure.

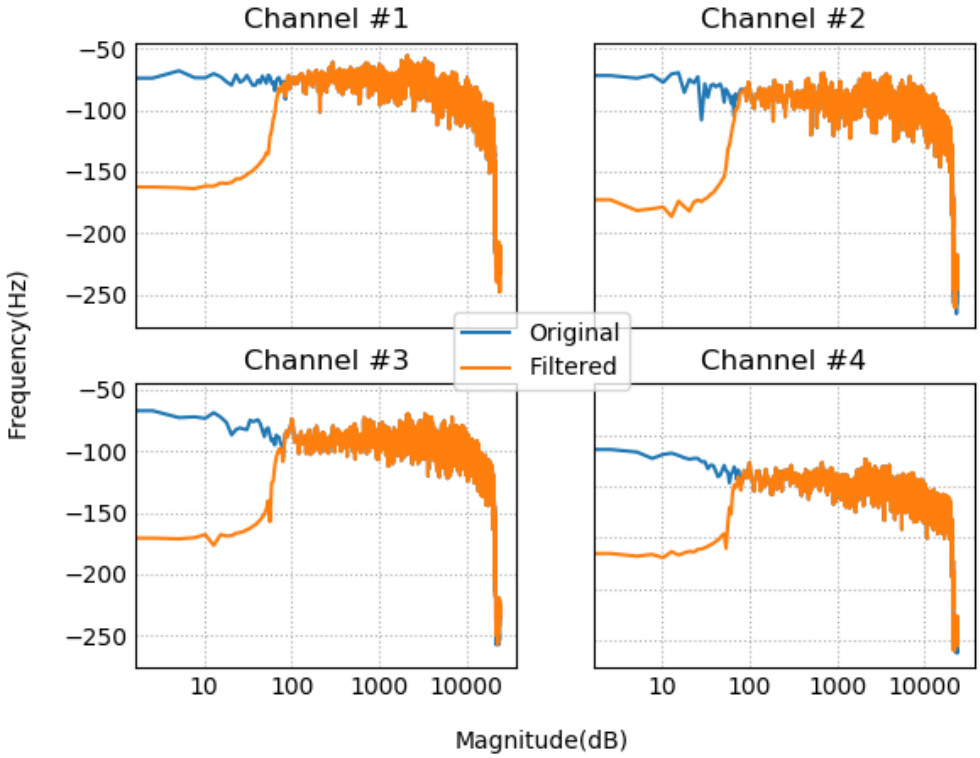
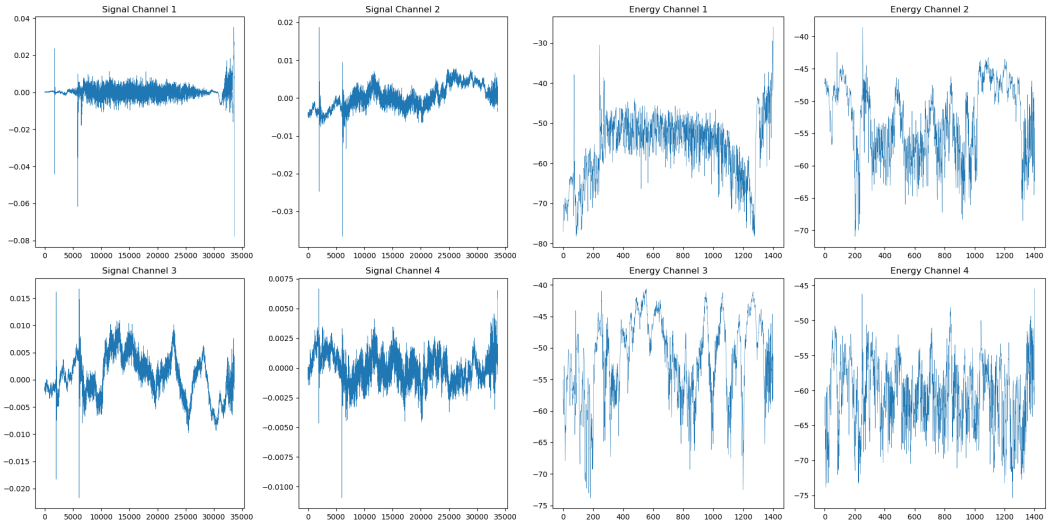
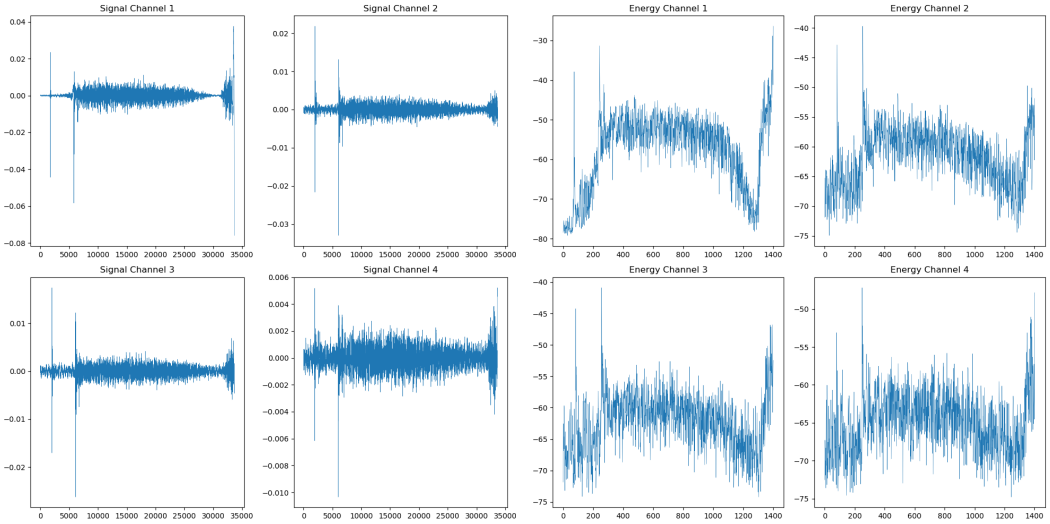


Figure 3.8: Effect of high-pass filtering

Also, detailed energy and amplitude curves of the breath instances resulted in more general trends after filtering, as can be observed below.



(a) Amplitude Before Filtering (b) Energy Before Filtering



(c) Amplitude After Filtering (d) Energy After Filtering

Figure 3.9: Effects of Filtering on Breath Instances

After noise is removed, we syncronized data from all channels using cross-correlation, to remove the delays from sound propagation. This is necessary in order to prevent the classifiers used to adapt to delays in data. An example of detected amounts of delays can be seen in the following figure.

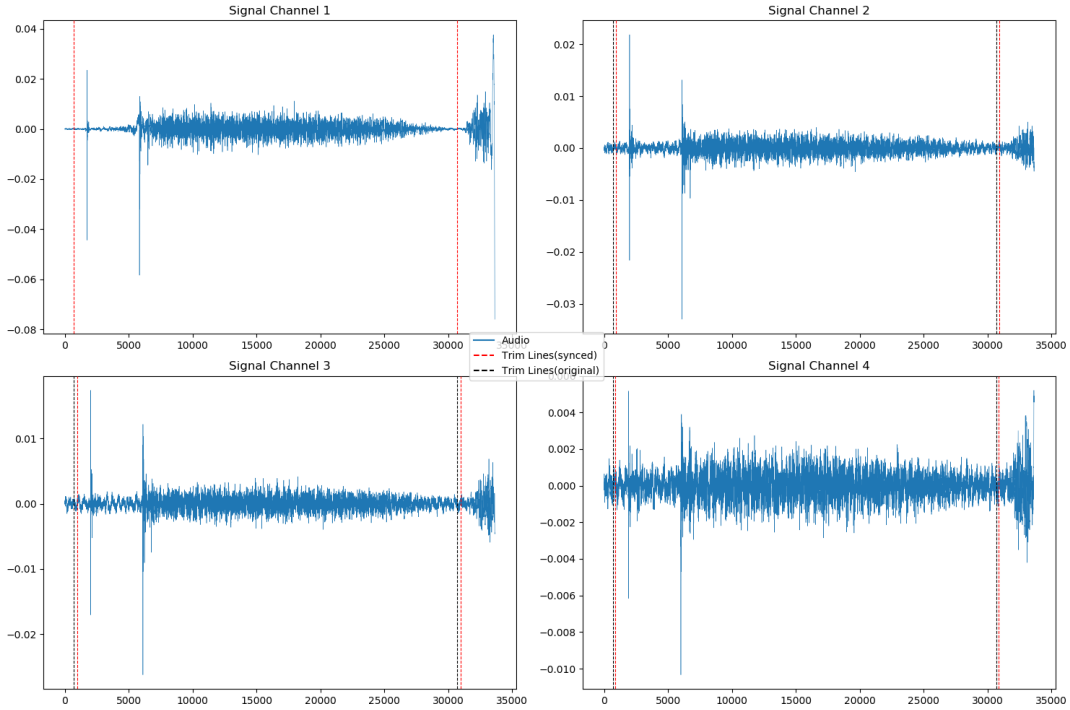


Figure 3.10: Detected Edges of Breath with Syncronization. The dotted lines indicate the range of delays

3.5.6 Length Analysis

When we analyzed our breath instances, we noticed that more than 90% of our data is shorter than 600 milliseconds. Using this characteristic, we implemented a parameter to shrink our data size for classification with more samples (explained in 5.2.2). Below is the graphic of length distribution of breath instances in the BreathBase.

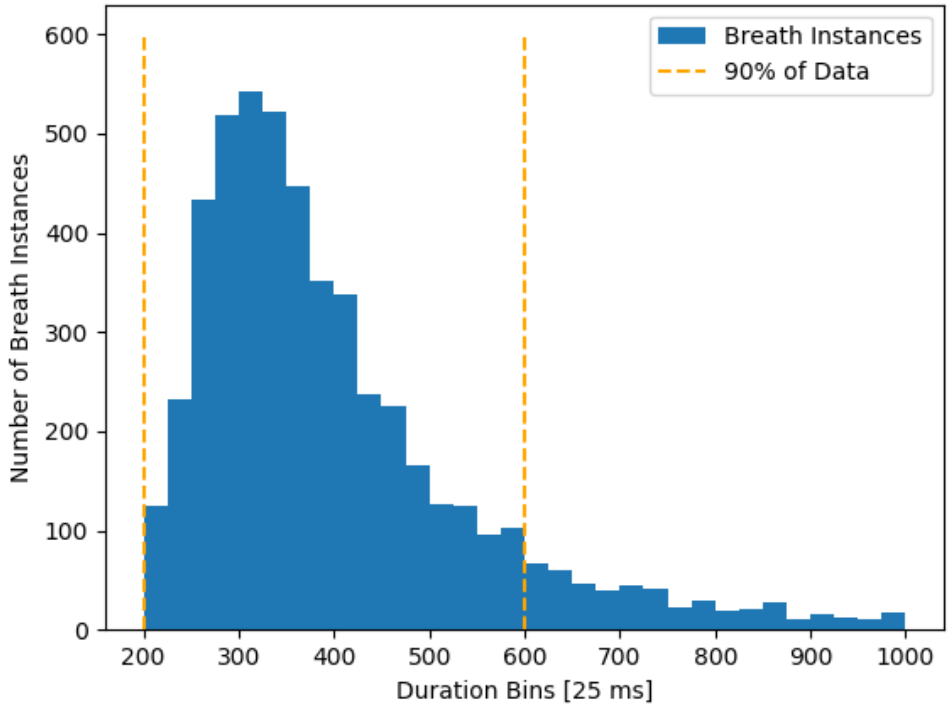


Figure 3.11: Length Analysis of BreathBase

3.5.7 Class Analysis

After analyzing the number of breath instances per each class, we noticed that all of our speaker classes have more than 100 examples (except one). This information let us to create a smaller version of the dataset with from instances from all speakers, to mature our models (explained in 5.2.2). Below is the graphic of class distribution of breath instances in the BreathBase. Please note that the minimum number of instances is 89, the maximum number of instances is 710 and the average for all classes is 253.5 breath instances.

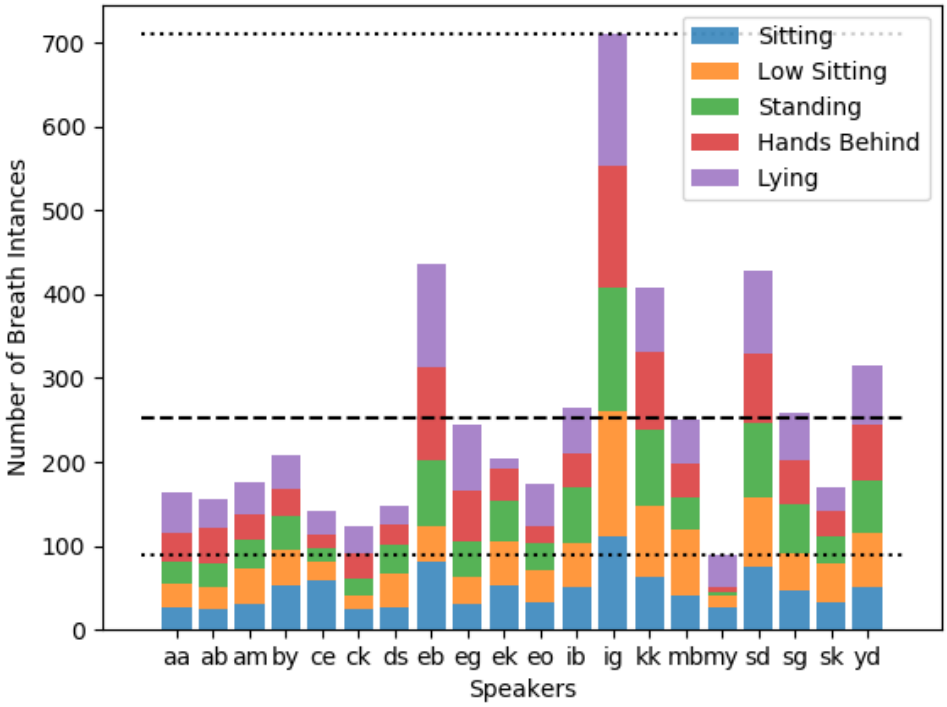


Figure 3.12: Class Analysis of BreathBase

CHAPTER 4

CLASSIFICATION OF INTRA-SPEECH BREATHING

In this chapter, we will be explaining our parameters for feature extraction and our classification models for the best use of these features.

4.1 Feature Modes

For our experiments, following instantaneous acoustic features were extracted from IMFs: 1. Instantaneous Frequency, 2. Unwrapped Phase and 3. Magnitude. These features, extracted from both normalized and initial versions of IMFs, were used for the experiments. Also, multiple selections from these features were concatenated to have a feature vector with a greater length (l_{vec}) to work with. Following options were implemented for the feature extraction phase:

- [Frequency] : $l_{vec} = n_{imf}$
- [Magnitude] : $l_{vec} = n_{imf}$
- [Phase] : $l_{vec} = n_{imf}$
- [Normalized Frequency] : $l_{vec} = n_{imf}$
- [Normalized Magnitude] : $l_{vec} = n_{imf}$
- [Normalized Phase] : $l_{vec} = n_{imf}$
- [Frequency + Magnitude] : $l_{vec} = n_{imf} \cdot 2$
- [Magnitude + Phase] : $l_{vec} = n_{imf} \cdot 2$
- [Frequency + Phase] : $l_{vec} = n_{imf} \cdot 2$
- [Frequency + Normalized Frequency] : $l_{vec} = n_{imf} \cdot 2$
- [Magnitude + Normalized Magnitude] : $l_{vec} = n_{imf} \cdot 2$
- [Phase + Normalized Phase] : $l_{vec} = n_{imf} \cdot 2$
- [Frequency + Magnitude + Phase] : $l_{vec} = n_{imf} \cdot 3$

4.2 Channel Modes

Recording channels were also used all at once in the feature vector or each channel separately, representing different recording conditions. There are multiple variations of these channel modes using the given features extracted:

- Separate and Front Channels:
  For the cases where each channel was treated as a separate recording condition and their classification rates were investigated, separate channel mode was used for any of the four channels. Another derivation from this mode was using only the channel recorded by a microphone directly 1.5 meters in front of the speaker. This option also selects one channel for each recording but, for recordings at sitting positions channel 2, standing positions channel 3 and lying positions channel 4 is selected as feature source. For these options feature matrix(M_{feat}) and input space(S_{inp}) are constructed using number of recordings(n_{rec}), number of time samples in the recording(n_{ts}) and extracted feature vector length (l_{vec}) as below:

$$\begin{aligned} M_{feat} &= n_{ts} \times l_{vec} \\ S_{inp} &= n_{rec} \times M_{feat} \end{aligned} \tag{4.2.1}$$

- Split Channels:
  For the cases where we needed to measure performance with more data, we used each channel as a separate recording. Since we have 4 different channels (n_{chn}), this approach increased the number of recordings by 3 times. This way, slightly different recordings are considered as augmented versions of the recordings with different conditions. For these options, feature matrix and input space are constructed as below:

$$\begin{aligned} M_{feat} &= n_{ts} \times l_{vec} \\ S_{inp} &= (n_{rec} \cdot n_{chn}) \times M_{feat} \end{aligned} \tag{4.2.2}$$

- Overlapped Channels:
  These modes were used for augmenting feature vector by increasing its length. We concatenated feature vectors from the next 3 samples after the sample's vector. This option was applied beside other channel modes as an alternative for investigating the effects of vector length. Although the number of time samples in a recording was shortened by 3, total feature matrix extended like below:

$$\begin{aligned} M_{feat} &= (n_{ts} - 3) \times (l_{vec} \cdot 4) \\ S_{inp} &= n_{rec} \times M_{feat} \end{aligned} \tag{4.2.3}$$

- All Channels:
  These variations were used to classify using also the information in-between channels along time-series. For each time step, feature vectors from all channels were concatenated into one vector and used accordingly.

$$\begin{aligned} M_{feat} &= n_{ts} \times (l_{vec} \cdot n_{chn}) \\ S_{inp} &= n_{rec} \times M_{feat} \end{aligned} \tag{4.2.4}$$

Although it did not affect the size of input space in any way, three different implementations of this channel mode, altering channels' orders while concatenation were also developed. In the regularly ordered variation, each channel was concatenated with the default order given in 3.5.2. For the randomized in unison variation, some randomized order of channels were applied to all recordings and features were concatenated with that same order along with all recordings. In the last variation, the channel concatenation order is randomized for each recording.

4.3 Classes

For our experiments, we classified recordings both by speaker identity and recording posture. For speaker classification, number of classes (n_{cls}) increased over time. Initial experiments (see chapter 5) started with 5 speakers, but at the complete dataset, there are 20 different participants. For the posture setting, we classified each of the postures differently by mapping them to 5 different classes and also grouped postures into 3 classes as sitting, standing and lying. High and low sitting positions are merged under one class, just as two standing positions.

4.4 Classification Methods

In this section, our classification algorithms are examined from different aspects with implemented variations and the architectures are explained with parts constructing the classifier itself. Please note that our classification process is divided into two phases for changes in data post-processing (refer to 3.5.5), classification architectures (refer to 4.4.6) and callback mechanisms (refer to 4.4.5). Methods were also implemented to minimize the effects of different step sizes (refer to 5.2) for fitting maximum data into limited memory.

4.4.1 Layers

4.4.1.1 Convolutional Layers

Convolutional networks are applicable to many types of data in array form and they exploit local correlations in the array of the data. Since we have continuous, time-series sound data, we anticipate that the feature vectors of neighboring time steps convey strong relations in-between. Convolutional layers were used to identify these correlations in a hierarchical manner and extract low to high levels of features.

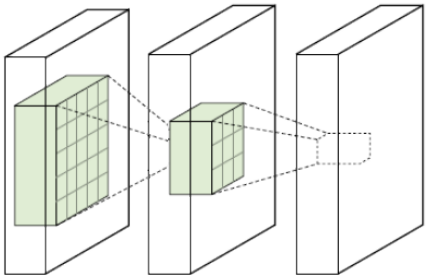


Figure 4.1: Two Layer 2D Convolution

4.4.1.2 Recurrent Layers

Convolutional networks model invariances across space, recurrent neural networks (RNN) do something similar across time using units to act as memories. Long Short-Term Memory (LSTM) [51] units are better alternatives for time-series classification problem because of preventing gradients from vanishing. LSTM layers were used to identify correlations on time domain, based on already dense high-level features extracted by the convolution layers, to support sequence classification. A simpler variant of LSTM units, Gated Recurrent Units (GRU) [52] was also used for its simplicity since we needed to fit larger data with computationally more efficient architectures due to memory limitations [53]. GRUs combine LSTM's forget and input gates into a single update gate and also merge the cell state and hidden state.

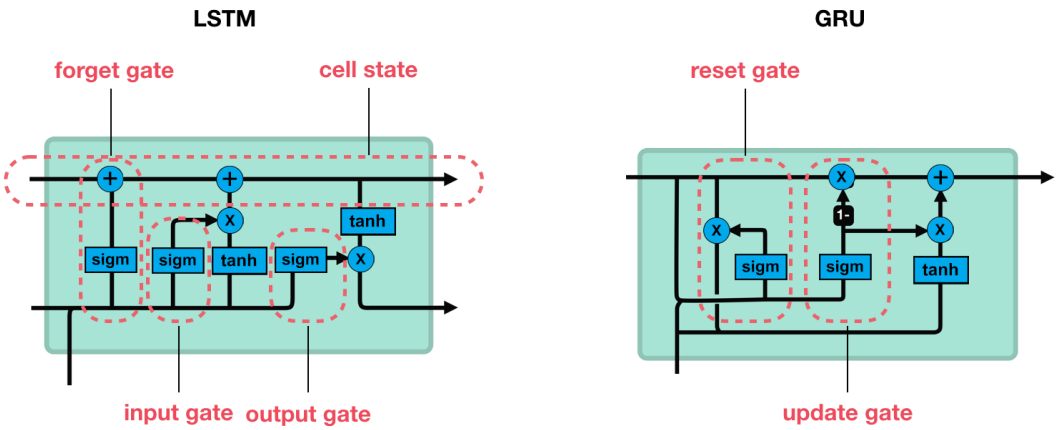


Figure 4.2: LSTM vs GRU

4.4.2 Optimization Methods

4.4.2.1 Stochastic Gradient Descent

Stochastic Gradient Descent algorithm implements a way to minimize an objective function $J(\theta)$ by updating model parameters (θ) according to learning rate (η), in opposite direction of the gradient. Generic update function of the algorithm (can be referred to as vanilla or batch gradient descent) is:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta) \tag{4.4.1}$$

where $\nabla_\theta J(\theta)$ is the gradient of the objective function with regards to the model parameters [54]. Among the variations of this algorithm, the difference is the amount of data used per update, in other words, when to update. Stochastic Gradient Descent calculates the update of the parameters of the network for a number of random inputs of each batch. Because of this approach, it works much faster than other variations (batch or mini-batch gradient descent) and allows online learning, but suffers from heavy gradient fluctuations because of the frequent updates. Update function of Stochastic Gradient Descent is:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i)}; y^{(i)}) \tag{4.4.2}$$

where $x^{(i)}$ and $y^{(i)}$ is each example's input-output pair.

4.4.2.2 SGD with Nesterov Accelerated Gradient

Stochastic Gradient Descent with Nesterov Momentum aims to mitigate the fluctuations of the SGD algorithm. The main idea is accumulating the previous steps' velocity vector and correcting the current update of gradient [55]. Nesterov momentum smooths the updating process and removes the fluctuations.

$$\begin{aligned} v_{t+1} &= \mu \cdot v - \eta \cdot \nabla_\theta J(\theta; x^{(i)}; y^{(i)}) \\ \theta_{t+1} &= \theta_t + v_{t+1} \end{aligned} \tag{4.4.3}$$

where μ is momentum coefficient, v_{t+1} is the velocity vector of current update and θ_{t+1} is the current gradient update.

4.4.2.3 Root Mean Square Propagation

Root Mean Square propagation (RMSProp) algorithm [56] is another derivative of gradient descent algorithm with adaptive learning rates. The algorithm computes the gradient of the batch by dividing it by a running average of its previous magnitudes

of recent gradients. So RMSProp uses derivates an exponentially decaying average to discard history from the extreme past.

$$\begin{aligned} g &= \nabla_\theta J(\theta; x^{(i)}; y^{(i)}) \\ r_{t+1} &= \rho r_t + (1-\rho) g \odot g \\ \theta_{t+1} &= \theta_t - \frac{\epsilon}{\delta + \sqrt{r}} \odot g \end{aligned} \tag{4.4.4}$$

where g is the current gradient, r_{t+1} is the gradient accumulation variable of current update, ρ is decay rate, ϵ is a small constant to stabilize the divison, θ_{t+1} is the current gradient update, and $\odot$ is element-wise multiplication operator.

4.4.2.4 Adaptive Moment Estimation

Adaptive Moment Estimation (Adam) [57] algorithm can be thought as a combination of RMSProp and Nesterov momentum with a few important distinctions. Adam uses the first-order moment of gradient and bias-corrected momenta.

$$\begin{aligned} g &= \nabla_\theta J(\theta; x^{(i)}; y^{(i)}) \\ s &= \rho_1 s_t + (1-\rho_1)\, g \\ r &= \rho_2 s_t + (1-\rho_2)\, g \odot g \\ s_{t+1} &= \frac{s}{1-\rho_1^t} \\ r_{t+1} &= \frac{r}{1-\rho_2^t} \\ \theta_{t+1} &= \theta_t - \epsilon \frac{s_{t+1}}{\sqrt{r_{t+1}} + \delta} \end{aligned} \tag{4.4.5}$$

where s is biased first moment estimate update, r is biased second moment estimate update, s_{t+1} is correction of first moment, r_{t+1} is correction of second moment θ_{t+1} is the current gradient update.

Also, Nadam(an exact combination of RMSProp and Nesterov momentum, excluding distinctions) and Adamax (where ρ is infinity so, Adam with infinity norm) variants of Adam algorithm were used in the experiments.

4.4.3 Loss Functions

Categorical cross-entropy is intended for use with multi-class classification where each class is assigned a unique value, from 1 to n. It is the preferred loss function under the inference framework of maximum likelihood. Cross-entropy calculates a

score that summarizes the average difference between the actual and predicted probability distributions for all classes in the problem. Aim of a network architecture is to minimize this loss and a perfect cross-entropy value is zero.

$$H(p, q) = \sum_i p_i \log \frac{1}{q_i}. \tag{4.4.6}$$

Kullback Leibler Divergence (KL Divergence) is a measure of how similar a probability distribution p is to a candidate distribution q. A KL divergence loss of zero suggests the distributions are identical. In practice, the behavior of KL Divergence is very similar to cross-entropy. It calculates how much information is lost if the predicted probability distribution is used to approximate the desired target probability distribution.

$$D_{\text{KL}}(p|q) = \sum_i p_i \log \frac{p_i}{q_i} \tag{4.4.7}$$

And from that, we can show the relationship in-between as below.

$$H(p, q) = H(p) + D_{\text{KL}}(p|q). \tag{4.4.8}$$

4.4.4 Activation Functions

Activation functions determine the output of a neuron in a neural network. These functions determine whether a node is activated or not, based on the neuron's input. Their outputs range between 0 to 1 or -1 to 1. Following activation functions (in the form of $y = f(x)$) were used during our experiments.

- Hyperbolic Tangent: $y = tanh(x)$
- Linear: $y = x$
- Rectified Linear Unit (ReLU): $y = max(0, x)$
- Sigmoid: $y = \frac{1}{1+e^{-x}}$
- Softmax: $y = \frac{x_i}{\sum x}$

4.4.5 Callback Functions

A set of callback functions are introduced to networks (in Phase 2) to be applied at given stages of the training procedure. The following functionalities are implemented.

- Model Checkpoint: After every epoch of the training, validation loss is calculated and the model with the least validation loss is saved (with its weights) for later use while ensembling models.
- Early Stopping: Training is stopped after 45 epochs of training without a validation loss drop more than 10^{-4}.

- Learning Rate Reduction: Learning rate is halved (up to 5 times) after 10 epochs of training without a validation loss drop more than 10^{-4}. After every learning rate reduction, a cooldown period of 10 epochs is also applied and calculations started after that.

4.4.6 Architectures: Phase 1

4.4.6.1 Initial Network: CNN-LSTM

Following initial network architecture was developed for exploring performances of different feature variations.

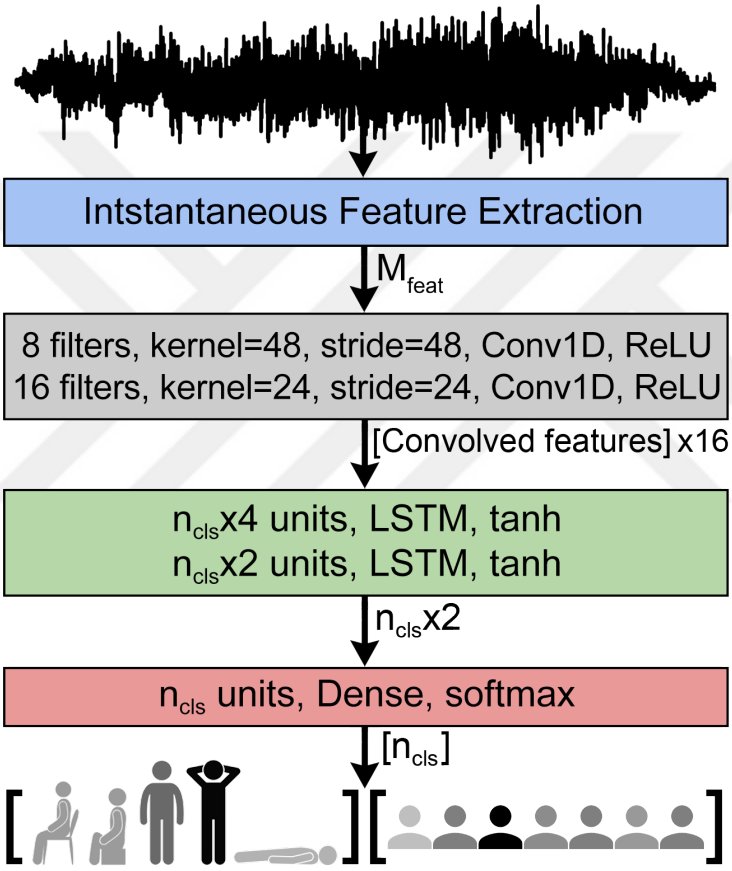


Figure 4.3: Network Archictecture[1]

In the input layer, extracted features from each time sample were fed to the network as a time-series data. These data were convolved with 8 filters with a kernel size of 48 timesteps (corresponding to 1 millisecond for 48khz sampling rate). Stride was given the same size as the kernel, so no overlapping occurs. Values read by 8 filters with the size of 48, at each 48 timestep updated weights with the application of ReLU. This function was chosen as a decision rule at the first layer to eliminate the unwanted

[1] Size of the convolved features differs depending on feature mode, channel mode and sampling rate

effects of possible negative values that came from the nature of the HHT, as described in section 2.3. Due to its simplicity (which affects computational cost) and ability to return absolute zero (instead of approximations to zero), this function was chosen.

In the second layer, the number of filters was doubled and kernel sizes halved. Thus, higher-level features were extracted in a more detailed fashion with more and smaller filters. No overlapping approach and the same activation function were kept for this layer.

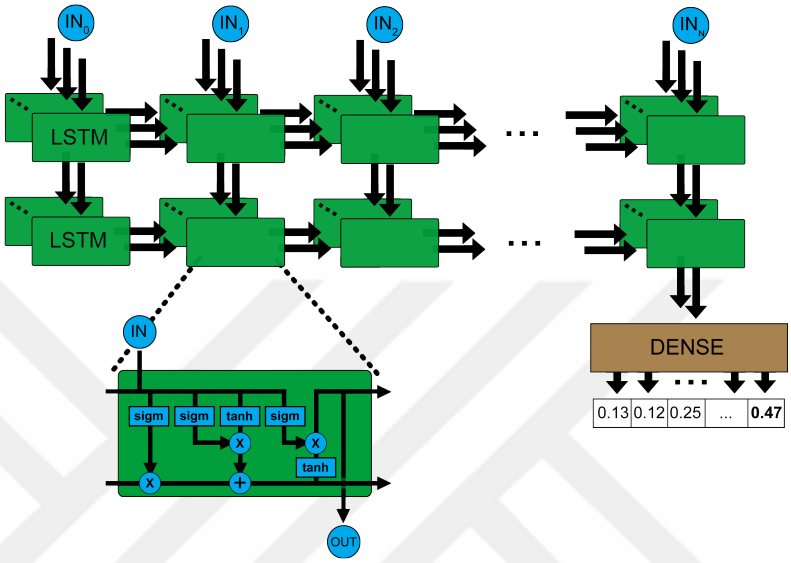


Figure 4.4: An example of 2 LSTM layers connected to a Dense layer

These higher-level features were fed into the first LSTM layers with $4n_{cls}$ units and for each input, $4n_{cls}$ sequences were passed to the second LSTM layer. Classification weights are gathered at the final step from $2n_{cls}$ units of LSTM layer and condensed with a softmax activation for acquiring the highest probable class in a one-hot vector form. The hyperbolic tangent activation function is the most commonly used activation function in LSTMs[58]. Since ReLU did not yield better results[59] and we did get the best results with hyperbolic tangent in initial experimentation (see chapter 5), we adopted this function in our network.

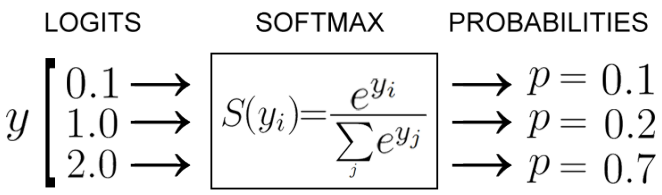


Figure 4.5: Softmax Activation Function

Softmax activation function was used in the Dense layer for getting probabilities of different classes and making a prediction based on that. Since softmax returns probabilities according to the rate of the exponential magnitude of an input value over the sum of all inputs, it is a well-suited option to choose.

4.4.6.2 Version 2: CNN-LSTM with Dropout

After examining the results of the first version of the classifier network, a second version of the network is implemented. In this version, 2 additional convolution layers were implemented to have better localization on data and 3 additional dropout layers with 0.5 drop rate is inserted in-between convolution layers to reduce the effects of the overfitting problem with increased localization. In this model, we also decreased the strides in convolutional layers to grasp more from time-series and to provide more information to LSTM layers.

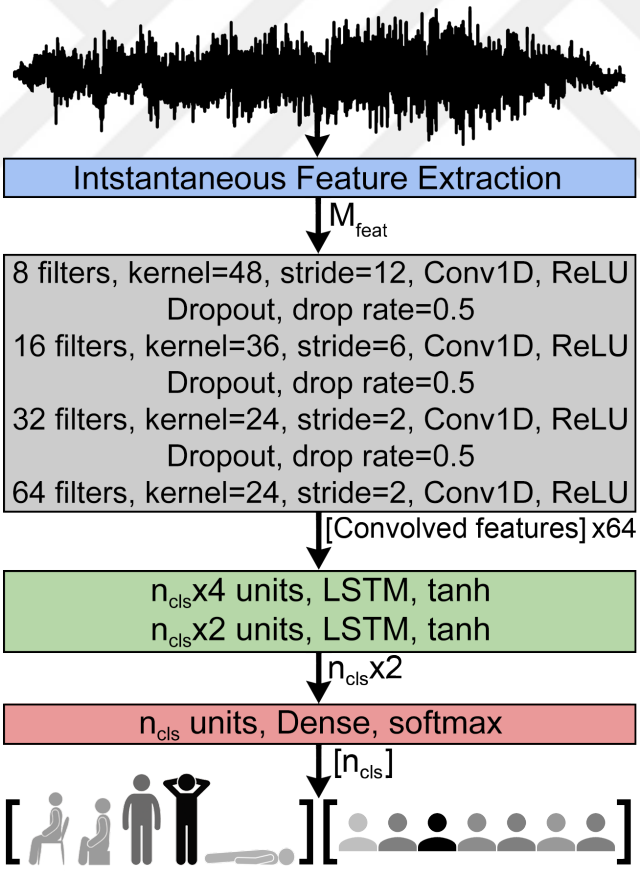


Figure 4.6: Network Archictecture: Version 2

4.4.7 Architectures: Phase 2

4.4.7.1 Version 3: CNN-GRU

After the second version of the network provided better results, we implemented a similar architecture with GRU layers, instead of LSTM ones. Since we had all the data collected and available for us in Phase 2, we needed a model with a lower computational cost. On the ensembling side, this model gathered information from high-level features with 4 layers of convolution but had issues with larger step sizes because of the same reason.

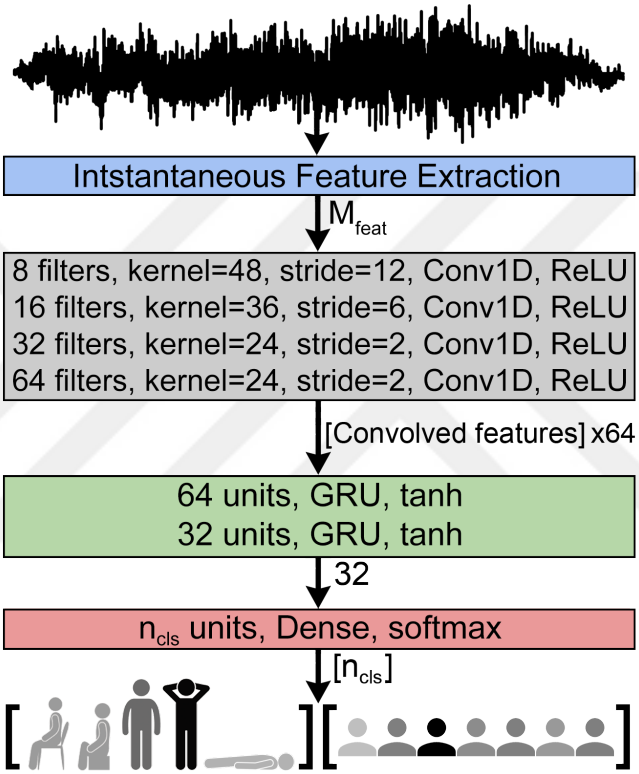


Figure 4.7: Network Archictecture: Version 3

4.4.7.2 Version 4: CNN-GRU, Balanced

Since the GRU counterpart of the model architecture yielded better results, we implemented a more balanced version of the previous model. Since the previous model suffers from larger step sizes, we dropped one convolution layer from architecture. We also increased the number of filters in each layer of convolution and inserted dropout layers in-between to reduce overfitting. On the ensembling side, this model learned from mid-level features of the data with filters with greater sizes and less number of convolutional layers.

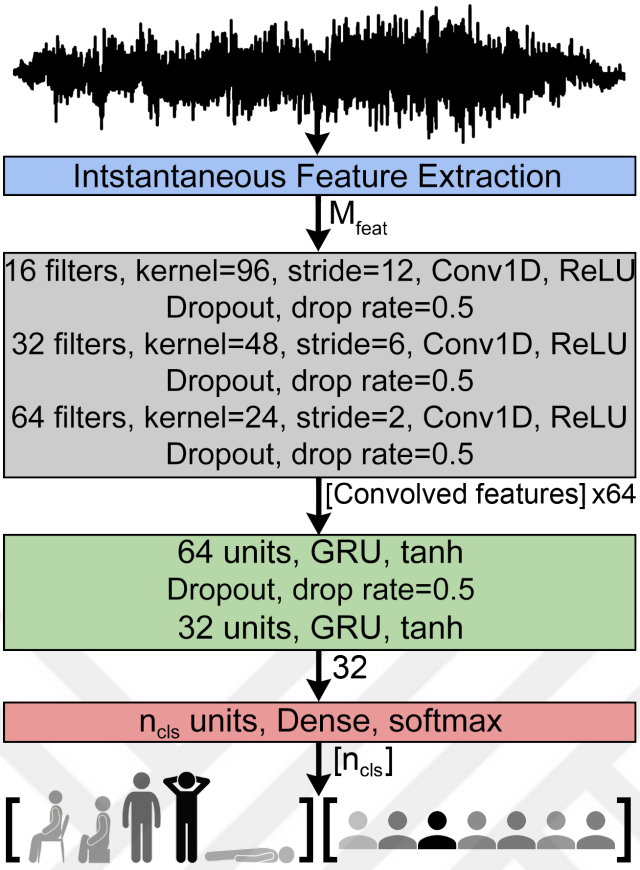


Figure 4.8: Network Archictecture: Version 4

4.4.7.3 Version 5: CNN-GRU, Less Dropout

We also implemented another version of the previous network with a lower drop rate for the better overall performance of ensembled models for variant step sizes.

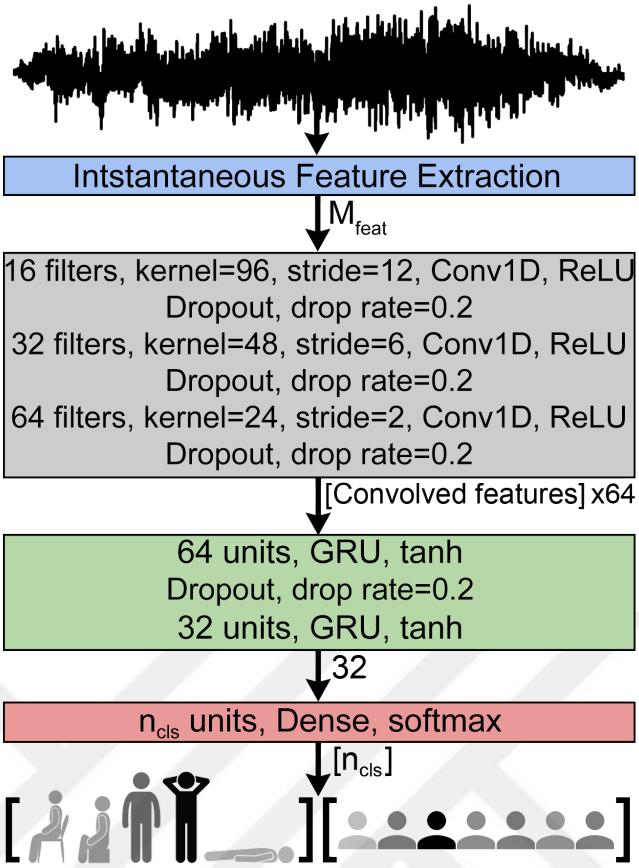


Figure 4.9: Network Archictecture: Version 5

4.4.7.4 Version 6: CNN-GRU, Simple Architecture

Another model is implemented for grasping low-level information from data. Since we have all instances available in Phase 2, enough training data can provide better resolution of relations in-between low-level features (especially after using all data instead of $allSmall$ set, as explained in 5.2.2). Also, for ensembling, this model learned from low-level features in data and grasped different connections from other models to increase variety.

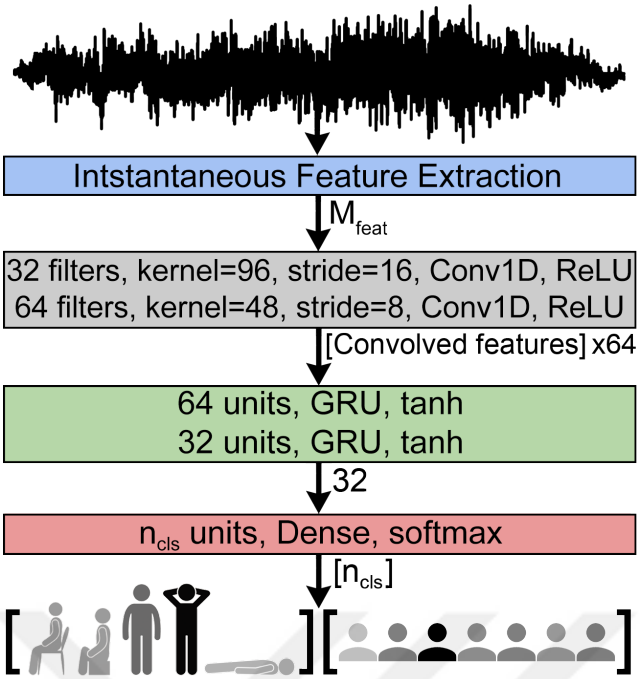


Figure 4.10: Network Archictecture: Version 6

4.4.7.5 Ensemble

For ensembling models, we put pre-trained models together and connected them to the same input layer. After the output of each model is generated, we took the resulting array's average to create ensembled probabilities for each class. By creating models aimed to grasp relations differently, we planned to improve our classification rate more than a single model can.

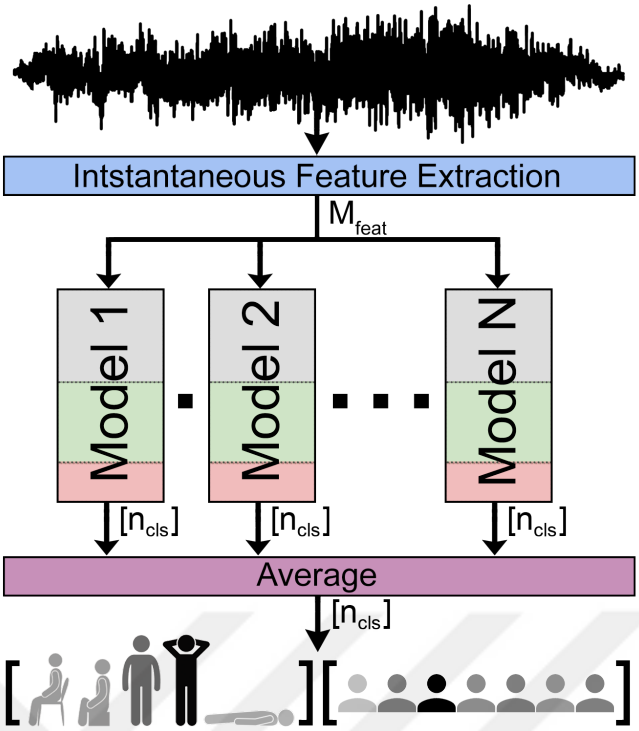


Figure 4.11: Network Archictecture: Ensembled

CHAPTER 5

EVALUATION

5.1 Evaluation of Dataset

Our dataset, BreathBase, was constructed from recordings of 20 participants, possessing more than 250 breath samples per-speaker (on average), recorded in a studio with a maximum background noise of 40 dB SPL and with professional recording equipment. It also provides tagging for 5 different postures and 4 different channels as different recording conditions for data variety. At the time of writing this thesis, there were no other datasets available neither for this purpose specifically nor as variationally rich. We consider our soon to be a publicly available dataset, BreathBase, as a contribution of our work.

5.2 Evaluation of Classification Method

Development of our main classification method, CNN-RNN algorithm started on earlier stages of data collection. The first successful run was completed on a 4 people dataset with nearly 800 breath instances, using pure Tensorflow API on GPU. All experiments conducted on a Windows 10 computer having 4-core 8-thread 2.8GHz CPU, 4GB GDDR5 GPU and 16GB 2400MHz RAM. Detailed specifications of the Nvidia GeForce GTX 1050 graphics card and its CUDA capabilities are listed below:

- CUDA Driver Version / Runtime Version: 9.1 / 9.0
- CUDA Capability Major/Minor version number: 6.1
- Total amount of global memory: 4096 MBytes (4294967296 bytes)
- 5 Multiprocessors, 128 CUDA Cores/MP: 640 CUDA Cores
- GPU Max Clock rate: 1493 MHz (1.49 GHz)
- Memory Clock rate: 3504 Mhz
- Memory Bus Width: 128-bit
- L2 Cache Size: 524288 bytes

Initial network reimplemented on Keras library using Tensorflow backend as an optimization. Even though it resulted in shorter epoch durations, after some expansion of the data in the dataset, the available GPU amount in the system could not keep up and *step size* parameter is used. This parameter used as stride or hop size to decide which samples to proceed to set the next time step. Naturally, if the step size is 1; for a data with length of 1000 samples, all 1000 samples are used for the experiments. On the other hand, if the step size is 4; for the same data, there are 250 timesteps. This means only the first one of each four consecutive samples is used and the next three is skipped. This parameter was used throughout the experiments and some alternatives were also implemented like reducing the sampling rate from 48kHz to 8kHz. But, there is a nuance between using a step size of 6 on a 48kHz input and reducing the input sampling rate to 8kHz. In the first option, if we have raw data with 600 samples, we first extract features from these 600 samples, then we skip some of them and use features of 100 samples. However, in the second option, we try to extract features from 100 samples, which results in less number of IMFs (see 2.3) and shorter feature vectors. This reduction of this data size means both good (less computational cost) and bad (fewer features to correlate), so both options were examined.

Also, for all experiments, 80% of the data was split for training, remaining data were used for validation. All accuracies stated below acquired on the validation sets of the current dataset at the time. These splits were done in a stratified manner, so each split had the same distribution. For the training phase, each mini-batch was also normalized (had the same distribution) to keep the covariate shift at a minimum.

A table summarizing configurations of experiments done is listed below.

Table 5.1: Configurations for Experiments

Reference Chapter	**Number of Participants**	**Sampling Rate**	**Step Size**	**Number of Epochs**	**Classification Networks**
5.2.1.1	4	48 kHz	4	300	v1
5.2.1.2	13	48 kHz	4	400, 500	v1
5.2.1.3	15	48 kHz	4	300	v1
5.2.1.4	15	48 kHz	6	300	v1
5.2.1.5	15	48 kHz	6	300	v1
5.2.1.6	15	48 kHz	4	300	v1
5.2.1.7	15	8 kHz	1	300	v1
5.2.1.8	15	8 kHz	1	200	v2
5.2.2.1	20	48 kHz	1, 2, 4	variable[2]	v3, v4, v5, v6
5.2.2.2	20	48 kHz	4, 8	variable[2]	v3, v4, v5, v6

[2]Because of the early stopping callback function (explained in 4.4.5) introduced in Phase 2, a pseudo big epoch number is given and model is trained until the stopping criteria is met.

5.2.1 Phase 1

5.2.1.1 Optimizer and Loss

For the first set of experiments, 760 breath recordings from 4 people were used. The following results were acquired for different optimizer and loss function variations using $All(regular)$ channel mode. Other details of the configuration are listed in Table 5.1.

Table 5.2: Results for: Optimizer and Loss

Optimizer		**Loss**	**Accuracy**	
Algorithm	**Learning Rate**	**Function**	**Magnitude**	**Frequency**
RMSProp	0.02	Cat. Cross Entropy	0.8449	0.3643
RMSProp	0.001	Cat. Cross Entropy	0.7132	0.4341
RMSProp	0.003	Cat. Cross Entropy	0.7364	0.4108
RMSProp	0.005	Cat. Cross Entropy	0.8372	0.4651
Adam	0.001	Cat. Cross Entropy	0.8837	0.4180
Adam	0.002	Cat. Cross Entropy	0.8527	0.4263
Adam	0.01	Cat. Cross Entropy	0.8450	0.4419
Adam	0.001	KL Divergence	0.7984	0.4573
SGD+Nest	0.01	Cat. Cross Entropy	0.3178	0.3178
SGD	0.01	Cat. Cross Entropy	0.3178	0.4109
Adamax	0.002	Cat. Cross Entropy	0.8372	0.3875
Nadam	0.002	Cat. Cross Entropy	0.8294	0.4186

After results in Table 5.2, *Adam optimizer with the learning rate of 0.001* and *categorical cross-entropy loss function* was chosen for further experiments. Experiments later on mostly intended to investigate effects of different feature and channel modes.

5.2.1.2 13 Participants

During the implementation of new feature modes, dataset expanded to 13 people with 2627 breath recordings and chosen network architecture is tested on this dataset using *All (regular)* channel mode for *speaker classification*. Other details of the configuration are listed in Table 5.1.

Table 5.3: Results for: 13 Participants

Feature Mode	Training Epoch	Classification Accuracy
Frequency	400	0.5736
Frequency	500	0.6356
Raw data[3]	500	0.3782

As an addition to the results in Table 5.3; same feature set extracted from the same dataset was fed to an SVM model with linear kernel and squared hinge loss, and 0.2533 accuracy was obtained.

5.2.1.3 Phase & Normalized

At the next step, feature space was enlarged and phase implementation was added with normalized versions of these feature modes. Channel modes were also investigated. After that, these variations are tried with a dataset of 15 people. Details of the configuration are listed in Table 5.1.

[3]Raw wave files were fed to network, their values(magnitudes at each timestep) as their feature vectors.

Table 5.4: Results for: Phase & Normalized

Channel Mode		0		1		2		3		Front	
Feat. Mode	**Classify By**	Speaker	Posture5	Speaker	Posture5	Speaker	Posture5	Speaker	Posture5	Speaker	Posture5
Frequency		0.177	0.2459	0.1803	0.259	0.159	0.2213	0.1885	0.2426	0.1869	0.218
Magnitude		0.4836	0.2754	0.3984	0.3803	0.3672	0.418	0.341	0.3541	0.4246	0.4426
Phase		0.1672	0.2393	0.1607	0.2262	0.1934	0.2344	0.1525	0.2492	0.1787	0.223
Frequency + Magnitude		0.177	0.2213	0.1689	0.2623	0.182	0.2197	0.182	0.2492	0.1852	0.2361
Magnitude + Phase		0.1852	0.2344	0.1656	0.2639	0.1607	0.2311	0.1967	0.241	0.1689	0.2311
Frequency + Phase		0.1705	0.2574	0.1885	0.2279	0.1541	0.2525	0.1902	0.2344	0.1672	0.2262
norm Frequency		0.1869	0.2557	0.1836	0.241	0.1689	0.2393	0.1705	0.2262	0.1754	0.2328
norm Magnitude		0.159	0.2328	0.1459	0.223	0.2833	0.2262	0.1672	0.2164	0.1623	0.2016
norm Phase		0.1787	0.2459	0.1754	0.241	0.1885	0.2164	0.1689	0.2541	0.177	0.2377
Freq + norm Freq		0.1574	0.2623	0.1787	0.2426	0.1557	0.241	0.1721	0.2541	0.177	0.2426
Mag + norm Mag		0.3049	0.2049	0.1443	0.2443	0.2492	0.2803	0.177	0.2295	0.159	0.1787
Phase + norm Phase		0.1607	0.2619	0.1656	0.2377	0.1574	0.2279	0.182	0.2311	0.1754	0.2787
Raw Data		0.3754	0.2508	0.2852	0.3344	0.2508	0.3738	0.2443	0.2951	0.2984	0.4115

As seen in table 5.4, IMF magnitudes yielded the best results.

5.2.1.4 Larger Channel Modes

For channel modes that result in larger feature vectors, only feature modes with shorter vector sizes were experimented. Even though, *step size* parameter had to be increased slightly. Other details of the configuration are listed in Table 5.1.

Table 5.5: Results for: Larger Channel Modes

Channel Mode		**Split**		**0-Overlapped**	
Feature Mode	**Classify By**	**Speaker**	**Posture5**	**Speaker**	**Posture5**
Frequency		0.1732	0.2524	0.1689	0.2279
Magnitude		0.4268	0.3562	0.4852	0.3164
Phase		0.176	0.254	0.1689	0.2262
		1-Overlapped		**2-Overlapped**	
Frequency		0.1869	0.2393	0.1869	0.2492
Magnitude		0.4443	0.4082	0.441	0.4213
Phase		0.1852	0.2164	0.182	0.241
		3-Overlapped		**All (regular)**	
Frequency		0.177	0.2328	0.1738	0.2148
Magnitude		0.4	0.3803	0.5689	0.7525
Phase		0.177	0.2557	0.1525	0.2321

5.2.1.5 All Channel Variations

After seemingly great results in Table 5.5 for *All* channel mode for posture classification, we implemented shuffling options for channels and compared results in-between. Details of the configuration are listed in Table 5.1.

Table 5.6: Results for: All Channel Variations

Channel Mode		**All (regular)**	
Feature Mode	**Classify By**	**Speaker**	**Posture5**
Magnitude		0.5689	0.7525
		All Shuffled Unison	
		Speaker	**Posture5**
Magnitude		0.5262	0.6934
		All Shuffled Random	
		Speaker	**Posture5**
Magnitude		0.4902	0.318

Results in the table 5.6 showed that when we concatenate channels in the feature vector, the order of the channels affects the result, drastically. Especially when channels are ordered differently for each input file, accuracy dropped radically. As a result of this behavior, we considered that the network might be learning delay durations for each microphone, instead of the breath characteristics. Since the speed of the sound and microphone locations were identical and every participant speaks from nearly in the same positions for the same postures, this might be a probable reason for this difference between different channel orders. We addressed this concern in Phase 2 (see 5.2.2).

5.2.1.6 3 Posture Classification

After these implications, we examined classifying capabilities with three main posture classes; sitting, standing and lying. The first two classes both had support from two different positions, so distributions between classes were unbalanced. Details of the configuration are listed in Table 5.1.

Table 5.7: Results for: 3 Posture Classification

Channel Mode	**0**	**1**	**2**	**3**	**Front**
Frequency	0.3852	0.3787	0.4164	0.4082	0.3885
Magnitude	0.4574	0.5443	0.5984	0.5328	0.6361
Phase	0.3525	0.4164	0.4098	0.4197	0.4115

5.2.1.7 8 kHz Downsampled

After results in Table 5.7, we implemented another alternative to the memory problem for training like stated *in the beginning of the chapter*, we downsampled our data to 8kHz. By losing from the number of extracted IMFs (see 5.2), we gained on memory and worked without the step size parameter (or $stepsize = 1$ which has no effect). Other details of the configuration are listed in Table 5.1.

Table 5.8: Results for: 8 kHz Downsampled (1/2)

Channel Mode		**0**			**1**			**2**			**3**		
Feat. Mode	**Cls. By**	Spkr	Pos5	Pos3	Spkr	Pos5	Pos3	Spkr	Pos5	Pos3	Spkr	Pos5	Pos3
Frequency		0.2348	0.2118	0.3941	0.1823	0.2348	0.3957	0.1478	0.2315	0.3941	0.1806	0.2677	0.399
Magnitude		0.3924	0.3087	0.445	0.3892	0.358	0.5468	0.3284	0.3645	0.5435	0.33	0.3629	0.5255
Phase		0.1724	0.2315	0.3941	0.1757	0.2414	0.3727	0.1773	0.1938	0.376	0.1642	0.2299	0.3793
norm Frequency		0.1511	0.243	0.3695	0.1823	0.22	0.3645	0.1938	0.2562	0.4154	0.179	0.2397	0.3892
norm Magnitude		0.1379	0.2118	0.3547	0.1445	0.2184	0.3547	0.1642	0.1806	0.3218	0.1642	0.2003	0.3645
norm Phase		0.1609	0.2381	0.4138	0.1823	0.2282	0.3596	0.1494	0.2282	0.3859	0.1675	0.243	0.3859
		Front			**Split**			**O - Overlapped**			**1 - Overlapped**		
Frequency		0.1741	0.2282	0.3842	0.1724	0.2496	0.3822	0.1806	0.2315	0.3696	0.1757	0.225	0.3612
Magnitude		0.3251	0.4105	0.5435	0.3851	0.3116	0.4901	0.3924	0.3268	0.4762	0.4056	0.3695	0.6141
Phase		0.156	0.2496	0.3777	0.179	0.2672	0.3929	0.1806	0.243	0.3974	0.1576	0.2611	0.3645
norm Frequency		0.1823	0.2365	0.3777	0.188	0.2389	0.4007	0.1773	0.2479	0.3711	0.1854	0.2365	0.3957
norm Magnitude		0.1609	0.1806	0.3859	0.2278	0.3009	0.4618	0.1823	0.2315	0.3481	0.156	0.2069	0.3629
norm Phase		0.1823	0.2184	0.4138	0.1839	0.2578	0.4068	0.1806	0.2562	0.3957	0.1757	0.243	0.3924

Table 5.9: Results for: 8 kHz Downsampled (2/2)

Channel Mode		**2 - Overlapped**			**3 - Overlapped**			**All (regular)**			**All Shuffled Unison**		
Feat. Mode	**Cls. By**	Spkr	Pos5	Pos3	Spkr	Pos5	Pos3	Spkr	Pos5	Pos3	Spkr	Pos5	Pos3
Frequency		0.1675	0.2184	0.3793	0.1823	0.2365	0.3892	0.1609	0.2479	0.3957	0.1691	0.2167	0.3908
Magnitude		0.3596	0.3744	0.6125	0.335	0.3498	0.5353	0.5681	0.7652	0.9458	0.5074	0.7438	0.9491
Phase		0.1675	0.2414	0.3678	0.1593	0.2496	0.4236	0.1691	0.2397	0.3941	0.1593	0.2578	0.3859
norm Frequency		0.1675	0.2578	0.3924	0.1823	0.22	0.3908	0.1872	0.2217	0.3941	0.1593	0.2529	0.3793
norm Magnitude		0.1461	0.2118	0.381	0.1642	0.2167	0.3612	0.1412	0.202	0.3711	0.1527	0.2003	0.3481
norm Phase		0.1544	0.2397	0.376	0.1544	0.2135	0.3957	0.1642	0.2315	0.3826	0.1609	0.2118	0.3662

	All Shuffled Random		
Frequency	0.1544	0.2282	0.3842
Magnitude	0.4204	0.2989	0.5452
Phase	0.1954	0.2365	0.4204
norm Frequency	0.1691	0.2512	0.3941
norm Magnitude	0.1626	0.1987	0.3727
norm Phase	0.1675	0.2463	0.3514

When we compared the results between different sampling rates (from tables 5.4, 5.5, 5.7, 5.4, 5.8 and 5.9), following table occured with most significant differences:

Table 5.10: Results for 8 kHz Downsampled: Top Different Accuracies

Channel Mode	Front	0 - Overlapped	Front	0
Feature Mode	Magnitude	Magnitude	Magnitude	Magnitude
Classify By	Speaker	Speaker	Posture3	Speaker
Accuracy (48kHz)	0.4246	0.4852	0.6361	0.4836
Accuracy (8kHz)	0.3251	0.3924	0.5435	0.3924
Difference	0.0995	0.0928	0.0926	0.0912

5.2.1.8 Network Architecure v2

After we examined all results gathered with both sampling rates, we noticed a gap between speaker classifying experiments using IMF magnitudes as features. Accordingly, three of the four top different results were doing speaker classification and all four of them were using IMF magnitudes as features. Since we know that correlation exists in the data from 48kHz experiments, we aimed towards constructing a better classifier network to experiment on the downsampled version of the dataset.

After second version of the network (see 4.6) was implemented, following results were acquired using *first channel* and *IMF magnitudes* for *speaker* classification. Other details of the configuration are listed in Table 5.1.

Table 5.11: Result Comparison for: Network Architecure v2

Network Version	**Number of Epochs**	**Accuracy**
v1	300	0.3924
v2	200	0.5468
Difference	100	0.1544

Results gathered from the second generation of the network showed 39% better results from the initial one, even 13% better than the 48kHz sampled version (0.4836). It also converged earlier than the initial network.

5.2.2 Phase 2

After data collection was completed, we implemented a few modifications on data and classifier algorithms, because of the concerns that we had in section 5.2.1.5 and our memory limitations.

On the data side, we synced all breath recordings from all channels in the time domain, using cross-correlation (refer to 3.10) to cancel out the probability of the network, learning delays for posture classification. While doing this alignment for each channel, we also noticed that there was a significant amount of noise generated at lower frequencies in recordings (explained in 3.5.5). We applied a high-pass filter to remove the noise from all channels.

On the algorithm side, we implemented memory optimizations for handling data arrays to use different views of the same array to limit memory allocations for vast chunks of data. These optimizations enabled us to work with more data with different channel modes and feature modes. Also, we gathered some statistics for our data (refer to 3.11) and figured out that more than 90% of our data is shorter than 600 milliseconds. Since we are padding all data to the length of the longest sample in the input space (1000 milliseconds), we noticed that we could benefit from this information before feeding data to the network. So, we introduced the *MaxLen* parameter, to crop longer sequences. We set the value of the parameter to 600 ms as mentioned, and gained 40% in size for compensation of losing a small amount of data for longer breath instances.

5.2.2.1 AllSmall Dataset

With optimizations in section 5.2.2, we were able to run our tests on a small dataset including 100 samples (except one, refer to 3.11) from each of the 20 speaker without any downsampling or frame skipping. We matured our models using this portion of the dataset with *All (regular)* channel mode, *IMF magnitudes* as features and *batch size* of 64; and shared our best results below. Other details of the configuration are listed in Table 5.1.

Table 5.12: Top Results for: AllSmall Dataset

Network Version	**Classify By**	**Step Size**	**Accuracy**
v4	Posture3	2	**0.9749**
v4	Posture5	2	**0.8141**
v4	Speaker	4	**0.7186**

5.2.2.2 Ensembled Models

After the results we achieved in section 5.2.2.1, we aimed to feed our models with all samples for better classification. To utilize the whole set of breath instances, we needed to apply a few more tuning. For running tests on all recordings in the dataset, we needed to shrink our examples for sure. Since we were losing some number of IMFs when we downsample 5.2, we proceeded with using the $stepsize$ parameter, minimized at 4, skipping 3 of every 4 frames. We also implemented our models considering durability to step size changes, and tested against that. Since we needed to gather most from limited data, we needed to implement more complex models. However, this was another bottleneck due to memory limitations. For this problem, we developed different models handling features from possibly different levels of the data. After we trained each model, we ensembled them to cover different levels of features. With this solution, we achieved to get results from the whole dataset with *step sizes* of 4 and 8, *IMF magnitudes* as features and *batch size* of 64; and shared our best results below. Other details of the configuration are listed in Table 5.1.

Table 5.13: Top Results for: Ensembled Models

Channel Mode	**Network Version**[4]	**Classify By**	**Step Size**	**Respective Accuracy**	**Ensembled Accuracy**
All (regular)	Ensembled (4, 5, 6)	Posture3	8	**0.9822**, 0.9813, 0.9783	0.9793
	Ensembled (4, 5, 6)	Posture5	8	0.8245, 0.8136, **0.8708**	0.8521
	Ensembled (4, 5, 6)	Speaker	8	0.8314, 0.7821, 0.8343	**0.8698**
All Shuffled Random	Ensembled (4, 5, 6)	Posture3	8	0.8511, 0.6529, 0.7367	**0.8550**
	Ensembled (4, 5, 6)	Posture5	8	0.4832, 0.5158, 0.5385	**0.5533**
	Ensembled (4, 5, 6)	Speaker	8	0.6844, 0.6529, 0.6874	**0.7377**
0	Ensembled (4, 5, 6)	Posture3	8	**0.5089**, 0.4724, 0.4744	0.4911
	Ensembled (4, 5, 6)	Posture5	8	0.3629, 0.3057, 0.3304	**0.3570**
	Ensembled (4, 5, 6)	Speaker	8	0.6144, 0.5602, 0.6026	**0.6321**
Split	Ensembled (4, 5, 6)	Posture3	8	0.6198, 0.5964, 0.5676	**0.6250**
	Ensembled (4, 5, 6)	Posture5	8	0.4371, 0.4273, 0.4117	**0.4514**
	Ensembled (4, 5, 6)	Speaker	8	0.6763, 0.68, 0.6509	**0.7194**

As seen in Table 5.13, *All (regular)* channel mode gave the best results. For posture classification, individual models yielded better scores, but for speaker classification, the ensembled model gave the best result among all experiments. Also, ensembled models tended to score higher when the $stepsize$ parameter was 8, instead of 4 for all channel modes.

As for *All Shuffled Random* channel mode, posture classification results were im-

[4]Please note that other combinations of ensembling models with different network architectures were also experimented, but top results were yielded by the same combination, consistently.

proved significantly (see Table 5.6) and the gap with *All (regular)* channel mode was narrowed. Speaker classification results were also higher than expected, 0.1124 lower than the best. We believe that our synchronization for delays (refer to section 3.5.5) cleared the concerns about classifier network, learning delays (see section 5.2.1.5). Further thoughts about the topic are given in section 5.4.

Among single-channel modes, the first channel (with the index 0) was best of all 4 channels. When we run our experiments with *Split* channel mode to utilize information from all channels as different inputs, we acquired 10% higher classification rates, and this was the best out of all single-channel modes. Although, these results were not matched to the experiments with *All (regular)* channel mode, especially for posture classification.

5.3 Evaluation of Features

For our experiments, we decided to stop the sifting process at the 9th IMF for our original recordings with 48khz sampling rate. For the downsampled versions with 8khz sampling rate, the 7th IMF was the stopping criterion. These are the maximum numbers of IMFs that we can sift from each breath signal in the dataset. Among all feature configurations, magnitudes extracted from IMFs are considered most promising. Also, posture classification using all channels concatenated, yielded remarkable results. Some of the most accurate results are summarized below.

Table 5.14: Accuracy of results from features

Classify By	Speaker	Posture5	Posture3
Sampling Rate	48 kHz	48 kHz	48kHz
Step Size	8	8	8
Channel Mode	All (regular)	All (regular)	All (regular)
Feature Mode	Magnitude	Magnitude	Magnitude
Network Version	Ensembled (4, 5, 6)	v6	v4
Confusion Matrix	Figure 5.1	Figure 5.2	Figure 5.3
Classification Report	Table 5.15	Table 5.16	Table 5.17
Accuracy	0.8698	0.8708	0.9822

Confusion matrices and classification results of these classifiers can be seen below:

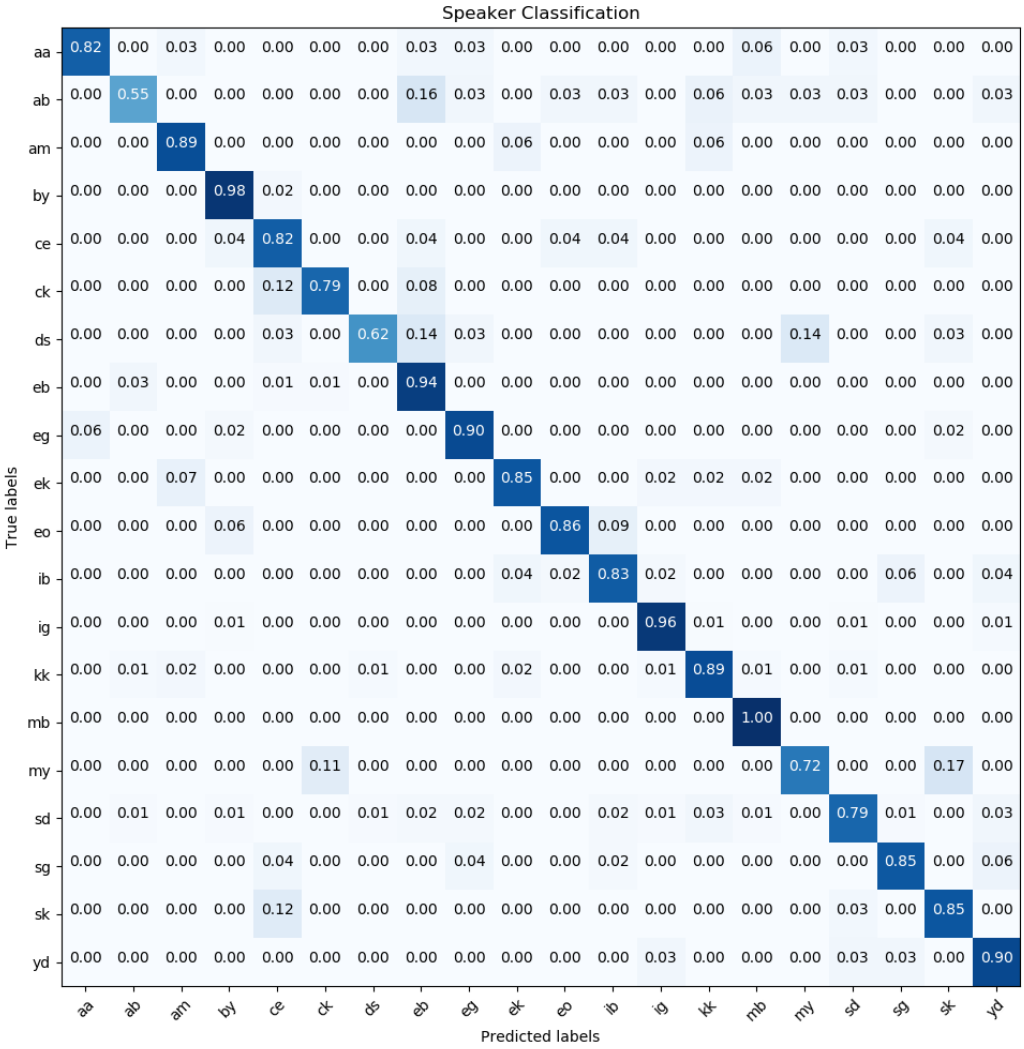


Figure 5.1: Confusion matrix for best speaker classification

Table 5.15: Classification results for best speaker classification

Label	Precision	Recall	f1 Score	Support
aa	0.90	0.82	0.86	33
ab	0.77	0.55	0.64	31
am	0.84	0.89	0.86	35
by	0.87	0.98	0.92	42
ce	0.66	0.82	0.73	28
ck	0.86	0.79	0.83	24
ds	0.90	0.62	0.73	29
eb	0.85	0.94	0.89	87
eg	0.86	0.90	0.88	49
ek	0.85	0.85	0.85	41
eo	0.91	0.86	0.88	35
ib	0.85	0.83	0.84	53
ig	0.96	0.96	0.96	142
kk	0.89	0.89	0.89	82
mb	0.89	1.00	0.94	50
my	0.72	0.72	0.72	18
sd	0.91	0.79	0.84	86
sg	0.88	0.85	0.86	52
sk	0.83	0.85	0.84	34
yd	0.84	0.90	0.87	63
micro avg	0.87	0.87	0.87	1014
macro avg	0.85	0.84	0.84	1014
weighted avg	0.87	0.87	0.87	1014

It can be seen from above results, there is no distinct confusion among specific classes, reciprocally. Although, it can be inferred that higher number of inputs per class yield better classification rate. This characteristic can be interpreted as the classifier needing more data -as number of inputs per class or number of features per input- for better speaker classification.

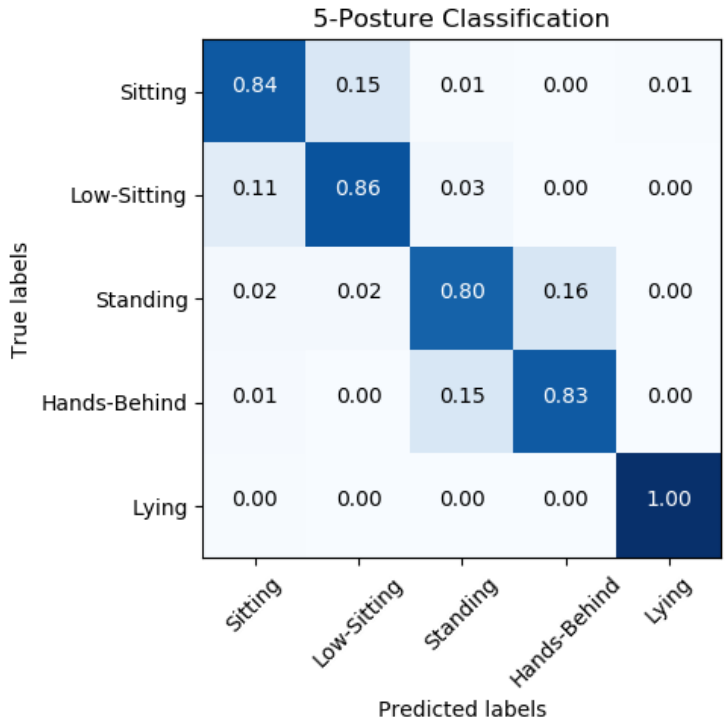


Figure 5.2: Confusion matrix for best 5-posture classification

Table 5.16: Classification results for best 5-posture classification

Label	**Precision**	**Recall**	**f1-score**	**Support**
Sitting	0.84	0.84	0.84	187
Low-Sitting	0.84	0.86	0.85	200
Standing	0.81	0.80	0.81	199
Hands-Behind	0.84	0.83	0.84	200
Lying	1.00	1.00	1.00	228
micro avg	0.87	0.87	0.87	1014
macro avg	0.87	0.87	0.87	1014
weighted avg	0.87	0.87	0.87	1014

Above results showed that two standing and sitting positions were confused among each other. But, lying position was distintly classified apart from other positions.

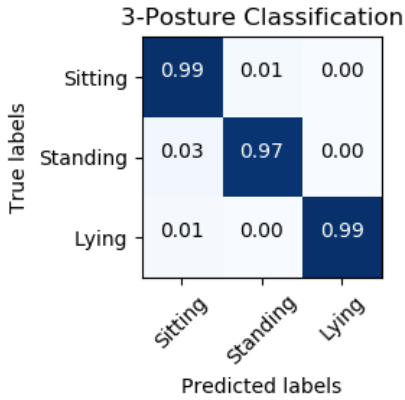


Figure 5.3: Confusion matrix for best 3-posture classification

Table 5.17: Classification results for best 3-posture classification

Label	**Precision**	**Recall**	**f1-score**	**Support**
Sitting	0.97	0.99	0.98	387
Standing	0.99	0.97	0.98	399
Lying	1.00	0.99	1.00	228
micro avg	0.98	0.98	0.98	1014
macro avg	0.98	0.98	0.98	1014
weighted avg	0.98	0.98	0.98	1014

Confusion between standing and sitting positions were resolved as these positions were merged into common sitting and standing positions. Although merge operation also caused some lying inputs to be classified as sitting, overall result was improved significantly.

5.4 Discussion

All records from all 20 participants of the dataset could not be utilized with the original sampling rate into the network because of memory problems. The increasing number of breath samples caused allocation errors. A memory upgrade or a different environment can be used to utilize all data into work, but for the moment, we have not had that chance. Experimenting with the algorithm on the whole dataset with 48kHz sampling rate and without any time sample skipped ($stepsize = 1$), should reveal all the potential of the dataset. Also, a third test set must be split for tests of the hyperparameter tuning. Tests on the complete dataset must include this split too.

Another point to consider is the batch size used for the experiments, which is not examined in Phase 1. In the second phase, we examined it for every model and got the best results with 64 instances per batch with a slight difference from 128. In phase 1, batch size even dropped to one sample to resolve memory issues, but the default parameter was

$$defaultBatchSize = \frac{totalNumberOfBreathInstances}{numberOfClasses}$$

for all experiments running without an error.

A run of the DTW-kNN algorithm [60, 61, 62] for speaker classification using 48Khz sampling rate, 200 breath instances of 15 people resulted in a 35% accuracy. We ran it with such a limited dataset because of the time it takes to predict, which increases exponentially as data grows. Since it was a limited version of the dataset, we believe that the algorithm needs some more tuning and improvement on computational cost for a direct comparison.

As another alternative, feeding spectrogram of the raw data into the classifier(s) was considered to be a baseline method to grade the features that we extracted but, it also returned results around the chance of a random choice, so another classification algorithm should be implemented for that kind of tests. Also, because Fourier transform outputs data with different length than input data, same principles did not apply for the exact comparison of a network.

We also noticed that for posture classification overfitting occurs less than speaker classification, when working with the whole dataset. This behavior may depend on higher number of samples per class. We ran our experiments with *split* channel mode to increase the input size but, since we lost in the length of the feature vector, experiments did not result in greater accuracies (though resulted better than single-channel modes), although a specially designed model can improve the accuracy of the mentioned configuration.

For accuracy of classification of both speakers or postures, we inferred that; the number of features extracted from a time sample (length of the feature vector, l_{vec}) outweighed the number of inputs fed to the network (n_{rec}) and the number of time samples in an input. Although our various feature modes concatenated frequency, magnitude and phase; these features yielded better results individually, meaning that they do not correlate with each other. On the other hand, concatenating the same feature from all channels resulted better than feeding channels individually. Also, the highest sampling rate (smallest *step size* parameter) did not always give the best results. This can be interpreted as IMFs with higher frequency (first IMFs extracted) convey a weaker relationship to the speaker. So that when we drop frequency, we do not lose correlation as much. With a higher number of correlated features extracted from each time sample, higher classification accuracy can be achieved.

Among all channels, the first channel, which was placed next to participants' mouth, yielded the best results. Other microphones, placed at 1.5 meters, yielded worse results, similar to each other. Although probable reason behind this can be different microphone selections (see chapter 3.5.1), this can also be interpreted as breath sig-

nals (along with IMFs and extracted features, inherently) are losing its specificity as the sound propagates. Similar reasoning can be made for the normalization of IMFs. When we applied normalization to IMFs, we lost specificity and classification results degraded.

Another topic to discuss was interchannel relationships for recordings. Since we recorded with 4 channels, using these channels together to enlarge our feature vector, was not an issue for us. But we consider that, for the applicability of our methods, this recording environment might not be feasible or even possible in every situation. In fact, this was the reasoning when we implemented different channel modes to assess the effects of using multiple channels at once for the classification. For posture classification with *All (regular)* channel mode, our inference was that our classifier network was learning postures in accordance with microphone positions. To cancel out this information, we mixed channels randomly for each input before feeding them into our classifier with *All Shuffled Random* channel mode. We implemented this channel mode, knowing that we also lost some of the integrity of data in the feature vector and made it harder for the classifier network to resolve correlations in time-domain. To keep information stable in time-domain, we used channels individually and fed each channel separately as different inputs with *Split* channel mode. Although our post-processing (see chapter 3.5.5) and optimizations in phase 2 of experiments removed the effects of delays and narrowed down the gap in-between, both alternative channel modes were not superior to the *All (regular)* channel mode. This meant that recording from different locations with multiple microphones has an impact on posture classification. But, alternative channel modes and model variations indicated that with more data and better-utilized models, similar accuracies can be achieved. Interchannel relationships only helped to achieve more with fewer data and less utilization.

Another thing to mention about these alternative channel modes was that *All Shuffled Random* channel mode resulted in higher accuracies than *Split* channel mode. Since the first one was favoring the length of the feature vector and the second one was favoring the number of inputs, these results also supported our claims of *number of features outweighing number of inputs* in the sixth paragraph of this section.

CHAPTER 6

CONCLUSION

As a conclusion of this thesis, we developed multiple variations of CNN-RNN networks to discover the potential of the higher dimensional acoustic features extracted from the breath. We consider our experiments successful to show the potential of the data and the features, with 87% speaker recognition accuracy in 20 classes and with 98% posture recognition accuracy in 3 classes (87% in 5 classes). We also explored various *features*, *channel modes*, *feature modes*, *hyperparameters* and *sampling rates*.

With all the experiments we completed during this thesis, we indicated that *IMF magnitudes* extracted from breath carry a strong relationship with the speaker. Also, multiple simultaneous recordings helped to improve this specificity, especially for posture classification. We believe our methods can provide useful side-information as speaker identification mechanism, independent of the content and the language. They also can increase the situation awareness for cases of emergency, as a decision support system.

As future work, our experiments can be expanded to address people with medical conditions. Our methods can be used as an indication mechanism of the diagnosis of respiratory diseases. They can also yield information about the sleeping stages and mechanics of the person, with additional breathing statistics. Our extracted features and statistical breathing information can also be used for recognition of a human's emotional state and well-being.

Also, our feature space can be extended to extract more information related to the field of study. We also created the BreathBase dataset for further studies in the field, currently only dataset developed specifically for this purpose.

REFERENCES

[1] L. Lu, L. Liu, M. J. Hussain, and Y. Liu, "I sense you by breath: Speaker recognition via breath biometrics," *IEEE Transactions on Dependable and Secure Computing*, 2017.

[2] W. Zhao, Y. Gao, and R. Singh, "Speaker identification from the sound of the human breath," *arXiv preprint arXiv:1712.00171*, 2017.

[3] A. K. Singh, *Handbook of Multimedia Information Security: Techniques and Applications*. Springer, 2019.

[4] M. Igras and B. Ziólko, "Wavelet method for breath detection in audio signals," in *2013 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, IEEE, 2013.

[5] T. Nakano, "Analysis and automatic detection of breath sounds in unaccompanied singing voice," *Proceedings of the 10th International Conference on Music Perception and Cognition (ICMPC10)*, pp. 387–390, 2008.

[6] D. Ruinskiy and Y. Lavner, "An algorithm for accurate breath detection in speech and song signals," in *2006 IEEE 24th Convention of Electrical & Electronics Engineers in Israel*, pp. 315–319, IEEE, 2006.

[7] D. Ruinskiy and Y. Lavner, "An effective algorithm for automatic detection and exact demarcation of breath sounds in speech and song signals," *IEEE transactions on audio, speech, and language processing*, vol. 15, no. 3, pp. 838–850, 2007.

[8] S. H. Dumpala and K. R. Alluri, "An algorithm for detection of breath sounds in spontaneous speech with application to speaker recognition," in *International Conference on Speech and Computer*, pp. 98–108, Springer, 2017.

[9] R. C. Sá and Y. Verbandt, "Automated breath detection on long-duration signals using feedforward backpropagation artificial neural networks," *IEEE transactions on biomedical engineering*, vol. 49, no. 10, pp. 1130–1141, 2002.

[10] T. Rosenwein, E. Dafna, A. Tarasiuk, and Y. Zigel, "Detection of breathing sounds during sleep using non-contact audio recordings," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 1489–1492, IEEE, 2014.

[11] A. Cohen and D. Landsberg, "Analysis and automatic classification of breath sounds," *IEEE Transactions on Biomedical Engineering*, no. 9, pp. 585–590, 1984.

[12] J. Chauhan, Y. Hu, S. Seneviratne, A. Misra, A. Seneviratne, and Y. Lee, "Breathprint: Breathing acoustics-based user authentication," in *Proceedings*

of the 15th Annual International Conference on Mobile Systems, Applications, and Services, pp. 278–291, ACM, 2017.

[13] A. S. Karunajeewa, U. R. Abeyratne, and C. Hukins, "Silence–breathing–snore classification from snore-related sounds," *Physiological Measurement*, vol. 29, no. 2, p. 227, 2008.

[14] M. Igras and B. Ziółko, "Different types of pauses as a source of information for biometry," *Models and analysis of vocal emissions for biomedical applications*, pp. 197–200, 2013.

[15] A. Janicki, "On the impact of non-speech sounds on speaker recognition," in *International Conference on Text, Speech and Dialogue*, pp. 566–572, Springer, 2012.

[16] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech communication*, vol. 52, no. 1, pp. 12–40, 2010.

[17] M. Igras-Cybulska, B. Ziółko, P. Żelasko, and M. Witkowski, "Structure of pauses in speech in the context of speaker verification and classification of speech type," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2016, no. 1, p. 18, 2016.

[18] V. Rapcan, S. D'Arcy, and R. B. Reilly, "Automatic breath sound detection and removal for cognitive studies of speech and language," *IET Irish Signals and Systems Conference*, 2009.

[19] P. Żelasko, T. Jadczyk, and B. Ziółko, "Hmm-based breath and filled pauses elimination in asr," in *2014 International Conference on Signal Processing and Multimedia Applications (SIGMAP)*, pp. 255–260, IEEE, 2014.

[20] T. Fukuda, O. Ichikawa, and M. Nishimura, "Detecting breathing sounds in realistic japanese telephone conversations and its application to automatic speech recognition," *Speech Communication*, vol. 98, pp. 95–103, 2018.

[21] I. Homma and Y. Masaoka, "Breathing rhythms and emotions," *Experimental physiology*, vol. 93, no. 9, pp. 1011–1021, 2008.

[22] S. Bloch, M. Lemeignan, and N. Aguilera-T, "Specific respiratory patterns distinguish among human basic emotions," *International Journal of Psychophysiology*, vol. 11, no. 2, pp. 141–154, 1991.

[23] T. Fukuda, O. Ichikawa, and M. Nishimura, "Breath-detection-based telephony speech phrasing," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[24] D. S. Avalur, "Human breath detection using a microphone," *Master of Science, Department of Mathematics and Natural Sciences, University of Groningen*, 2013.

[25] T. Emoto, M. Kashihara, U. R. Abeyratne, I. Kawata, O. Jinnouchi, M. Akutagawa, S. Konaka, *et al.*, "Signal shape feature for automatic snore and breathing sounds classification," *Physiological measurement*, vol. 35, no. 12, p. 2489, 2014.

[26] C. M. Rembold and P. M. Suratt, "Airway turbulence and changes in upper airway hydraulic diameter can be estimated from the intensity of high frequency inspiratory sounds in sleeping adults," *The Journal of physiology*, vol. 592, no. 17, pp. 3831–3839, 2014.

[27] K. S. Rao and A. K. Vuppala, *Speech processing in mobile environments*. Springer, 2014.

[28] J. R. Deller Jr, J. G. Proakis, and J. H. Hansen, *Discrete time processing of speech signals*. Prentice Hall PTR, 1993.

[29] J. Benesty, M. M. Sondhi, and Y. Huang, *Springer handbook of speech processing*. Springer, 2007.

[30] Z. Tüske, F. R. Drepper, and R. Schlüter, "Non-stationary signal processing and its application in speech recognition," in *SAPA-SCALE Conference*, 2012.

[31] M. I. Abdalla, H. M. Abobakr, and T. S. Gaafar, "Dwt and mfccs based feature extraction methods for isolated word recognition," *International Journal of Computer Applications*, vol. 69, no. 20, 2013.

[32] N. Gavriely and D. W. Cugell, "Airflow effects on amplitude and spectral content of normal breath sounds," *Journal of applied physiology*, vol. 80, no. 1, pp. 5–13, 1996.

[33] N. Gavriely, *Breath sounds methodology*. cRc PrEss, 2019.

[34] A. Sovijärvi, L. Malmberg, G. Charbonneau, J. Vanderschoot, F. Dalmasso, C. Sacco, M. Rossi, and J. Earis, "Characteristic of breath sounds and adventitious respiratory sounds," *Eur Respir Rev*, vol. 10, pp. 591–596, 01 2000.

[35] S. Reichert, R. Gass, C. Brandt, and E. Andres, "Pulmonary auscultation in the era of evidence-based medicine," *Revue des maladies respiratoires*, vol. 25, no. 6, pp. 674–682, 2008.

[36] E. Andres *et al.*, "Respiratory sounds analysis in the world of health 2.0 and medicine 2.0," *EC Pulmonology and Respiratory Medicine*, vol. 7, pp. 564–585, 2018.

[37] M. K. I. Molla and K. Hirose, "Single-mixture audio source separation by subspace decomposition of hilbert spectrum," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 3, pp. 893–900, 2007.

[38] H. Huang and X.-x. Chen, "Speech formant frequency estimation based on hilbert-huang transform," *JOURNAL-ZHEJIANG UNIVERSITY ENGINEERING SCIENCE*, vol. 40, no. 11, p. 1926, 2006.

[39] W. Wang, X. Li, and R. Zhang, "Speech detection based on hilbert-huang transform," in *First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS'06)*, vol. 1, pp. 290–293, IEEE, 2006.

[40] Z. Yang, D. Huang, and L. Yang, "A novel pitch period detection algorithm based on hilbert-huang transform," in *Chinese Conference on Biometric Recognition*, pp. 586–593, Springer, 2004.

[41] Z. Lu, B. Liu, and L. Shen, "Speech endpoint detection in strong noisy environment based on the hilbert-huang transform," in *2009 International Conference on Mechatronics and Automation*, pp. 4322–4326, IEEE, 2009.

[42] Z.-F. Liu, Z.-P. Liao, and E.-F. Sang, "Speech enhancement based on hilbert-huang transform," in *2005 International Conference on Machine Learning and Cybernetics*, vol. 8, pp. 4908–4912, IEEE, 2005.

[43] K. Khaldi, A.-O. Boudraa, M. Turki, T. Chonavel, and I. Samaali, "Audio encoding based on the empirical mode decomposition," in *2009 17th European Signal Processing Conference*, pp. 924–928, IEEE, 2009.

[44] N. E. Huang and S. S. Shen, *Hilbert-Huang Transform and Its Applications*, vol. 5. World Scientific, 2005.

[45] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[46] G. Wang, X.-Y. Chen, F.-L. Qiao, Z. Wu, and N. E. Huang, "On intrinsic mode function," *Advances in Adaptive Data Analysis*, vol. 2, no. 03, pp. 277–293, 2010.

[47] N. Huang and S. Long, "Normalized hilbert transform and instantaneous frequency," *NASA Patent Pending GSC*, vol. 14, pp. 673–680, 2003.

[48] N. E. Huang, "Introduction to the hilbert–huang transform and its related mathematical problems," in *Hilbert–Huang transform and its applications*, pp. 1–26, World Scientific, 2014.

[49] N. E. Huang, Z. Wu, S. R. Long, K. C. Arnold, X. Chen, and K. Blank, "On instantaneous frequency," *Advances in adaptive data analysis*, vol. 1, no. 02, pp. 177–229, 2009.

[50] M. Kleiner, *Electroacoustics*. CRC Press, 2013.

[51] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[52] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[53] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[54] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[55] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*, pp. 1139–1147, 2013.

[56] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning, lecture 6a, overview of mini-batch gradient descent," 2012.

[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[58] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, pp. 1310–1318, 2013.

[59] Q. V. Le, N. Jaitly, and G. E. Hinton, "A simple way to initialize recurrent networks of rectified linear units," *arXiv preprint arXiv:1504.00941*, 2015.

[60] A. Bagnall and J. Lines, "An experimental evaluation of nearest neighbour time series classification," *arXiv preprint arXiv:1406.4757*, 2014.

[61] N. Kulkarni, "Effect of dynamic time warping using different distance measures on time series classification," *International Journal of Computer Applications*, vol. 975, p. 8887, 2017.

[62] Y. Kim and E. M. Provost, "Emotion classification via utterance-level dynamics: A pattern-based approach to characterizing affective expressions," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3677–3681, IEEE, 2013.