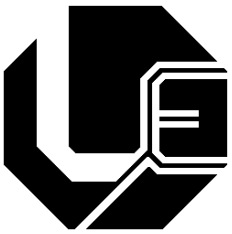
UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA



# A NOVEL WORD BOUNDARY DETECTOR BASED ON THE TEAGER ENERGY OPERATOR FOR AUTOMATIC SPEECH RECOGNITION

IGOR SANTOS PERETTA

UBERLÂNDIA
2010

IGOR SANTOS PERETTA

# A NOVEL WORD BOUNDARY DETECTOR BASED ON THE TEAGER ENERGY OPERATOR FOR AUTOMATIC SPEECH RECOGNITION

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Uberlândia, como requisito parcial para a obtenção do título de Mestre em Ciências.

Área de concentração: Processamento da Informação, Inteligência Artificial

Orientador: Prof. Dr. Keiji Yamanaka

UBERLÂNDIA

2010

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU , MG, Brasil

P437n Peretta, Igor Santos, 1974-
A novel word boundary detector base don the teager energy operator for automatic speech recognition [manuscrito] / Igor Santos Peretta. - 2010.
124 f. : il.

Orientador: Keiji Yamanaka.

Dissertação (mestrado) – Universidade Federal de Uberlândia, Programa de Pós-Graduação em Engenharia Elétrica.
Inclui bibliografia.

1. Reconhecimento automático da voz - Teses. 2. Redes neurais artificiais - Teses. I. Yamanaka, Keiji. II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

CDU: 681.3:007.52

IGOR SANTOS PERETTA

# A NOVEL WORD BOUNDARY DETECTOR BASED ON THE TEAGER ENERGY OPERATOR FOR AUTOMATIC SPEECH RECOGNITION

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Uberlândia, como requisito parcial para a obtenção do título de Mestre em Ciências.

Área de concentração: Processamento da Informação, Inteligência Artificial

Uberlândia, 21 de dezembro de 2010

Banca Examinadora

---

Keiji Yamanaka, PhD - FEELT/UFU

---

Hani Camille Yehia, PhD - DELT/UFMG

---

Gilberto Carrijo, PhD - FEELT/UFU

---

Shigueo Nomura, PhD - FEELT/UFU

À Anabela, minha esposa,
pelo amor, compreensão e
companheirismo.
À Isis, minha filha,
pelo amor e por todos
os sorrisos compartilhados.

# Agradecimentos

À minha esposa Anabela e à minha filha Isis, pelo amor, suporte, carinho e compreensão durante esta complicada fase de dedicação à minha pesquisa.

Aos meus pais, Vitor e Miriam, e aos meus irmãos, Érico e Éden, pelo amor e apoio incondicionais e por sempre acreditarem em mim.

Ao meu orientador, Prof. Dr. Keiji Yamanaka, pela confiança em mim depositada, pela presteza em auxiliar e pela grande oportunidade de trabalharmos juntos.

Aos meus companheiros em armas, Gerson Flávio Lima e Josimeire Tavares, pela amizade, companhia constante e pelos trabalhos realizados.

Ao Prof. Dr. José Roberto Camacho, pelas conversas, conhecimentos compartilhados e pelo espaço cedido para este trabalho.

Ao Prof. Dr. Shigueo Nomura, pelo suporte, análises e longas conversas.

Aos Prof. Dr. Gilberto Carrijo e Profa. Dra. Edna Flores, pelos conhecimentos compartilhados.

Ao Prof. Dr. Hani Camille Yehia, pelas considerações e contribuições inestimáveis.

Aos companheiros Eduardo, Marlus, Mattioli e Fábio, pela amizade, pelo café e pela ajuda imprescindível em momentos críticos.

Aos companheiros de laboratório, em especial a Élvio, Fabrício e Fernando, pelas conversas, auxílios e amizade.

Aos amigos Maria Estela Gomes, Pedro Paro, Edna Coloma e Lúcia Mansur, pelo apoio inigualável. Muito obrigado!

Aos amigos que deixei na DGA da Unicamp, pelo carinho, suporte e pelos momentos juntos. Em especial a Talita, Elza, Soninha, Renata, Marli, Angela, Serginho, Adagilson, Regina, Rozi, Ivone, Zanatta, Nanci, Pedro Henrique e Felipe.

Aos amigos de Campinas, Daltra & Dani, Barbá e Peri, pelas longas e saudosas conversas. À família Sérgio, Leandra, Miguel e Gabriela e à família Loregian, Fabiana e Tainá, pela amizade sincera.

Aos amigos de infância Carlos Augusto Dagnone e Marco Antônio Zanon Prince Rodrigues, pela longa amizade e pela caminhada que trilhamos juntos.

Ao Programa de Pós-Graduação da Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia, em especial aos Prof. Dr. Alexandre Cardoso, Prof. Dr. Edgard Lamounier e Cinara Mattos, pelo apoio e orientação nos diversos momentos desta pesquisa; também à agência CAPES, pela bolsa de incentivo à pesquisa concedida através deste programa.

*"We are all apprentices in a craft*
*where no one ever becomes a master."*
(ERNEST HEMINGWAY, 1961)

*"Somos todos aprendizes em um ofício*
*no qual ninguém nunca se torna mestre."*
(ERNEST HEMINGWAY, 1961)

# Resumo

Este trabalho é parte integrante de um projeto de pesquisa maior e contribui no desenvolvimento de um sistema de reconhecimento de voz independente de locutor para palavras isoladas, a partir de um vocabulário limitado. O presente trabalho propõe um novo método de detecção de fronteiras da palavra falada chamado "Método baseado em TEO para Isolamento de Palavra Falada"(TSWS). Baseado no Operador de Energia de Teager (TEO), o TSWS é apresentado e comparado com dois métodos de segmentação da fala amplamente utilizados: o método "Clássico", que usa cálculos de energia e taxa de cruzamento por zero, e o método "Bottom-up", baseado em conceitos de equalização de níveis adaptativos, detecção de pulsos de energia e ordenação de limites. O TSWS apresenta um aumento na precisão na detecção de limites da palavra falada quando comparado aos métodos Clássico (redu ção para 67,8% do erro) e *Bottom-up* (redução para 61,2% do erro).

Um sistema completo de reconhecimento de palavras faladas isoladas (SRPFI) também é apresentado. Este SRPFI utiliza coeficientes de Mel-Cepstrum (MFCC) como representação paramétrica do sinal de fala e uma rede *feed-forward* multicamada padrão (MLP) como reconhecedor. Dois conjuntos de testes foram conduzidos, um com um banco de dados de 50 palavras diferentes com o total de 10.350 pronúncias, e outro com um vocabulário menor — 17 palavras com o total de 3.519 pronúncias. Duas em cada três dessas pronúncias constituem o conjunto para treinamento para o SRPFI, e uma em cada três, o conjunto para testes. Os testes foram conduzidos para cada um dos métodos TSWS, Clássico ou *Bottom-up*, utilizados na fase de segmentação da fala do SRPFI. O TSWS permitiu com que o SRPFI atingisse 99,0% de sucesso em testes de generalização, contra 98,6% para os métodos Clássico e *Bottom-up*. Em seguida, foi artificialmente adicionado ruído branco gaussiano às entradas do SRPFI para atingir uma relação sinal/ruído de 15dB. A presença do ruído alterou a performance do SRPFI para 96,5%, 93,6% e 91,4% em testes de generalização bem sucedidos quando utilizados os métodos TSWS, Clássico e *Bottom-up*, respectivamente.

# Palavras-chave

Segmentação da Fala, Detecção de Fronteiras de Palavra Falada, TEO, Independente de Locutor, Palavras Isoladas, Sistema de Reconhecimento de Voz, MFCC, Redes Neurais Artificiais, MLP.

# Abstract

This work is part of a major research project and contributes into the development of a speaker-independent speech recognition system for isolated words from a limited vocabulary. It proposes a novel spoken word boundary detection method named "TEO-based method for Spoken Word Segmentation" (TSWS). Based on the Teager Energy Operator (TEO), the TSWS is presented and compared with two widely used speech segmentation methods: "Classical", that uses energy and zero-crossing rate computations, and "Bottom-up", based on the concepts of adaptive level equalization, energy pulse detection and endpoint ordering. The TSWS shows a great precision improvement on spoken word boundary detection when compared to Classical (67.8% of error reduction) and Bottom-up (61.2% of error reduction) methods.

A complete isolated spoken word recognition system (ISWRS) is also presented. This ISWRS uses Mel-frequency Cepstral Coefficients (MFCC) as the parametric representation of the speech signal, and a standard multilayer feed-forward network (MLP) as the recognizer. Two sets of tests were conducted, one with a database of 50 different words with a total of 10,350 utterances, and another with a smaller vocabulary — 17 words with a total of 3,519 utterances. Two in three of those utterances constituted the training set for the ISWRS, and one in three, the testing set. The tests were conducted for each of the TSWS, Classical or Bottom-up methods, used in the ISWRS speech segmentation stage. TSWS has enabled the ISWRS to achieve 99.0% of success on generalization tests, against 98.6% for Classical and Bottom-up methods. After, a white Gaussian noise was artificially added to ISWRS inputs to reach a signal-to-noise ratio of 15dB. The noise presence alters the ISWRS performances to 96.5%, 93.6%, and 91.4% on generalization tests when using TSWS, Classical and Bottom-up methods, respectively.

# Keywords

Speech Segmentation, Spoken Word Boundary Detection, TEO, Speaker-Independent, Isolated Words, Speech Recognition System, Mel-frequency Cepstral Coefficients, Artificial Neural Network, MLP.

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# Chapter 1

# Introduction

## 1.1 Overview

Speech interface to machines is a subject that has fascinated the humankind for decades. Let us put aside magical entities that respond to spoken commands like the *Golem* in Jewish tradition, or even the entrance of the cave in the *Ali Baba and the Forty Thieves* Arabic tale. In modern literature, we can track down to L. Frank Baum's *Tik-Tok* mechanical man from *Ozma of Oz* (1907) as the first machine that reacts to human spoken language. Recently, movies like *2001: A Space Odyssey* (1968) to the new released *Iron Man* (2008) and *Iron Man II* (2010) present us with computers that can comprehend human speech – *HAL* and *JARVIS*, respectively.

Engineers and scientists have been researching spoken language interfaces for almost six decades[1]. In addition to being a fascinating topic, speech interfaces are fast becoming a necessity. Advances in this technology are needed to

[1]We could detach the work of Davis, Biddulph and Balashek [15] as one of the first speech recognition systems.

enable the average citizen to interact with computers, robots, networks, and other technological devices using natural communication skills. As stated by Zue and Cole [74], "without fundamental advances in user-centered interfaces, a large portion of society will be prevented from participating in the age of information, resulting in further stratification of society and tragic loss of human potential". They also stated: "a speech interface, in a user's own language, is ideal because it is the most natural, flexible, efficient, and economical form of human communication".

Several different applications and technologies can make use of spoken input to computers. The conversion of a captured acoustic signal to a single command or a stream of words is the top of mind application for speech recognition, but we can also have applications for speaker's identity recognition, language spoken recognition or even emotion recognition.

After many years of research, speech recognition technology is generating practical and commercial applications. Some examples for English speech recognition could be found: Dragon Naturally Speaking; Project54 system; TAPTalk module; IBM ViaVoice; Microsoft SAPI among others. For Portuguese language, some commercial and freeware solutions are available. But, the common sense is that speech recognition softwares still roughly work as desired.

Why is so hard to reach an ultimate solution for speech recognition? Some languages are easier to reach an acceptable margin of successful recognition rates than others. Applications with smaller vocabulary size could perform better than large vocabulary ones. But, even solutions with acceptable recognition rates easily drop their rates down when immersed in high noise environments. Speaker-independent systems hardly contemplate enough diversification of utterances. Even speaker-dependent softwares have difficulties to identify speech if its user has some sort of temporally vocal dis-

order. The main reason is science still does not understand the full process of hearing: how exactly our brain translates acoustic signals to information after preprocessing of inner ear; how can we focus in a determined acoustic source in detriment of others; or even how do we easily understand corrupted acoustic information inside a familiar context. Those answers could not be provided, as far as we have searched for them.

## 1.2 Motivation

Table 1.1, extracted from Zue, Cole and Ward [75], shows the typical parameters used to characterize speech recognition systems.

Table 1.1: Parameters used to characterize speech recognition systems

| **Parameters** | **Range** |
|---|---|
| Speaking Mode | Isolated words to continuous speech |
| Speaking Style | Read speech to spontaneous speech |
| Enrollment | Speaker-dependent to Speaker-independent |
| Vocabulary | Small (< 20 words) to large (> 20,000 words) |
| Language Model | Finite-state to context-sensitive |
| Perplexity | Small (< 10) to large (> 100) |
| SNR | High (> 30 dB) to low (< 10 dB) |
| Transducer | Voice-canceling microphone to telephone |

This work is part of a research that aims to develop a speaker-independent speech recognition system for recognition of spontaneous spoken isolated words, with a not so small vocabulary, that could be embedded to several possible applications. This research intends to develop human-machine interfaces using Brazilian Portuguese language.

One of the most important aspects to this objective is to have a good speech segmentation algorithm. Speech segmentation is the core of speech recognition, because the recognizer needs to handle only with the speech

fragments of a given audio signal. This work proposes a novel speech segmentation method to support speech recognition and presents a complete recognition system implemented with widely used stages to achieve 99.0% of successful recognition rates on generalization tests.

## 1.3 Database

The adopted database for this project is the one constructed by Martins [42]. It is constituted of 50 words, as presented in Table 1.2, three utterances each, from 69 independent-speakers (46 men and 23 women, all adults). In this work, we chose to represent Brazilian Portuguese words pronunciation using the International Phonetic Alphabet (IPA). The IPA is an alphabetic system of phonetic notation and it has been devised by the International Phonetic Association as a standardized representation of all sounds in a given spoken language[2]. By convention, we present the phonemic words between slashes (/. . ./).

The captured audio signal for all the samples from this database had passed through a hardware[3] bandpass filter with cutoff frequencies (3dB) of 300Hz and 3,400Hz. The audio signals were captured with a sampling frequency of 8kHz and sample of 16 bits. For this work, all audio signals from this database were converted to *WAVEform Audio* format (*wav* file extension), and has their respective length extended to support speech segmentation testings. Also, they had white Gaussian noise added to each audio file, to reach a Signal to Noise Ratio (SNR) of 30dB (considered a high level SNR, i.e., a low level of noise compared to the signal).

[2]For more information on IPA, please consult the "Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet", Cambridge University Press, 1999.

[3]DSP-16 Data Acquisition Processor, from Ariel manufacturer.

Table 1.2: Brazilian Portuguese voice commands from the project database.

| ID | Command | English | IPA |
|---|---|---|---|
| 0 | ZERO | ZERO | /ˈzɛɾu/ |
| 1 | UM | ONE | /ˈũ/ |
| 2 | DOIS | TWO | /ˈdoyʒ/ |
| 3 | TRÊS | THREE | /ˈtreʒ/ |
| 4 | QUATRO | FOUR | /ˈkwatɾu/ |
| 5 | CINCO | FIVE | /ˈsĩku/ |
| 6 | SEIS | SIX | /ˈseyʒ/ |
| 7 | SETE | SEVEN | /ˈsɛtɪ/ |
| 8 | OITO | EIGHT | /ˈoytu/ |
| 9 | NOVE | NINE | /ˈnɷvɪ/ |
| 10 | MEIA[4] | HALF / SIX | /ˈmeya/ |
| 11 | SIM | YES | /ˈsĩ/ |
| 12 | NÃO | NO | /ˈnãw/ |
| 13 | TERMINAR | FINISH / QUIT | /teɾmiˈnaɾ / |
| 14 | REPETIR | REPEAT | /ɾepɪˈtiɾ/ |
| 15 | CONTINUAR | CONTINUE | /kõtinuˈaɾ/ |
| 16 | VOLTAR | BACK | /volˈtaɾ/ |
| 17 | AVANÇAR | FORWARD | /avãˈsaɾ/ |
| 18 | CERTO | CORRECT | /ˈsɛɾtu/ |
| 19 | ERRADO | WRONG | /eˈɾadu/ |
| 20 | OPÇÕES | OPTIONS | /opˈsõyʒ/ |
| 21 | DÓLAR | DOLLAR | /ˈdɷlaɾ/ |
| 22 | REAL | REAL | /ɾɪˈal/ |
| 23 | TEMPO | TIME / WEATHER | /ˈtẽpu/ |
| 24 | NORTE | NORTH | /ˈnɷɾtɪ/ |
| 25 | NORDESTE | NORTHEAST | /noɾˈdɛʒtɪ/ |
| 26 | SUL | SOUTH | /ˈsul/ |
| 27 | SUDESTE | SOUTHEAST | /suˈdɛʒtɪ/ |
| 28 | CENTRO-OESTE | MIDWEST | /sẽtroˈɛʒtɪ/ |
| 29 | ESPORTES | SPORTS | /ɪʒˈpɷɾtɪ/ |
| 30 | DEPARTAMENTO | DEPARTMENT | /depaɾtaˈmẽtu/ |
| 31 | DIVISÃO | DIVISION | /diviˈzãw/ |
| 32 | SEÇÃO | SECTION | /seˈsãw/ |
| 33 | COORDENAÇÃO | COORDINATION | /kooɾdenaˈsãw/ |
| 34 | IMAGEM | IMAGE | /iˈmajẽy/ |
| 35 | VOZ | VOICE | /ˈvɷʒ/ |
| 36 | ÁRIES | ARIES | /ˈarieʒ/ |
| 37 | TOURO | TAURUS | /ˈtowɾu/ |
| 38 | CÂNCER | CANCER | /ˈkãseɾ/ |
| 39 | LEÃO | LEO | /lɪˈãw/ |
| 40 | GÊMEOS | GEMINI | /ˈjemɪuʒ/ |
| 41 | VIRGEM | VIRGO | /ˈviɾjẽy/ |
| 42 | LIBRA | LIBRA | /ˈlibra/ |
| 43 | ESCORPIÃO | SCORPIO | /ɪʒkoɾpiˈãw/ |
| 44 | CAPRICÓRNIO | CAPRICORN | /kapriˈkɷɾniu/ |
| 45 | SAGITÁRIO | SAGITTARIUS | /sajiˈtaɾiu/ |
| 46 | AQUÁRIO | AQUARIUS | /aˈkwariu/ |
| 47 | PEIXES | PISCES | /ˈpeyxɪʒ/ |
| 48 | HORÓSCOPO | HOROSCOPE | /oˈɾɷʒkupu/ |
| 49 | AJUDA | HELP | /aˈjuda/ |

[4]In Brazilian Portuguese, “meia” is the same as “half”; here, it means “six” in reference to “half” dozen.

# Chapter 2

# State of Art

The history on Speech Recognition systems goes back to 1950's, when we can detach the work of Davis, Biddulph and Balashek [15] published in 1952. There, they have state "the recognizer discussed will automatically recognize telephone-quality digits spoken at normal speech rates by a single individual, with an accuracy varying between 97 and 99 percent" . Inside a low noise level controlled environment with a single speaker, they achieved an excellent recognition rate for "0" to "9" spoken digits.

From 1950's until now, several approaches have been discussed, mainly divided into three essential stages: speech segmentation, features extraction, and recognizer system. Besides the fact there are several intrinsic differences between several types of speech recognition systems, inspiring Zue et al. [75] to suggest parameters to label them (see Table 1.1), those three stages are essential to all of them. The following chapter intends to explore some of the widely used approaches to those stages.

## 2.1 Speech Segmentation

Speech segmentation could be defined as the boundary identification for words, syllables, or phonemes inside captured speech signals. As stated by Vidal and Marzal [69], "Automatic Segmentation of speech signals was considered both a prerequisite and a fairly easy task in most rather naïve, early research works on Automatic Speech Recognition". Actually, speech segmentation proved itself to be a complex task, specially when one is trying to segment speech as we segment letters from a printed text. Several algorithms were developed for the last decades. In the following paragraphs, some widely used segmentation speech, as some developed by recent works in the area, are presented.

The here named *Classical method* uses energy and zero-crossing rate computations [56], in order to detect the beginning and the ending of a spoken word inside a given audio signal. It is widely used until today, because it is relatively simple to implement.

Another method, proposed by Mermelstein [45], uses a convex-hull algorithm, to perform a "syllabification" of a spoken word inside a continuous speech approach, i.e., the segmentation of the speech syllable by syllable. Using an empirically determined loudness function, his algorithm "locates a boundary within the consonant roughly at the point of minimal first-formant frequency". According to Mermelstein, "[...] the syllabification resulting from use of the algorithm is generally consistent with our phonetic expectations".

*Bottom-up method*, also known as Hybrid Endpoint Detector, was proposed by Lamel, Rabiner, et al [35], based on concepts of adaptive level equalization, energy pulse detection and ordering of endpoints. They have defined their designed endpoint detector as consisting "of an optimized strat-

egy for finding endpoints using a three-pass approach in which energy pulses were located, edited, and the endpoint pairs scored in order of most likely candidates".

Zhou and Ji [73] have designed a real-time endpoint detection algorithm combining time-frequency domain. Their algorithm uses "the frequency spectrum entropy and the Short-term Energy Zero Value as the decision-making parameters". The presented results are not significant to enable comparisons.

Using regression fusion of boundary predictions, Mporas, Ganchev, and Fakotakis [47] have studied "the appropriateness of a number of linear and non-linear regression methods, employed on the task of speech segmentation, for combining multiple phonetic boundary predictions which are obtained through various segmentation engines". They were worried about the extraction of phonetic aligned in time, considered a difficult task. In their work, they have employed 112 speech segmentation engines based on hidden Markov models.

Some other works related to this subject could be found in the literature, as the work of Liu et al. [36], regarding the automatic boundary detection for sentences and disfluencies; the work of Yi and Yingle [71], that uses the measure of the stochastic part of a complexity movement to develop a robust speech endpoint detection in noisy environments; and the work of Ghaemmaghami et al. [20], with a method that utilizes gradient based edge detection algorithms, original from image processing field, to peform boundary detection for continuous speech in noisy conditions.

## 2.2 Feature Extraction

It is very important for any speech recognition system design to select the best parametric representation of acoustic data. This parametric representation is constituted of the features to be extracted from the speech signal. Some parametric representation starts from the study of how the speech is produced by human sound production system. Others starts from study of how the speech is perceived by human auditory system.

Figure 2.1 shows three different utterances from the same speaker for the word OPÇÕES /op'sõyʒ/. As one can see, those speech signals are more distinguishable between each other than we expect them to be. The right choice for the parametric representation is essential to minimize differences between several utterances of the same phoneme.

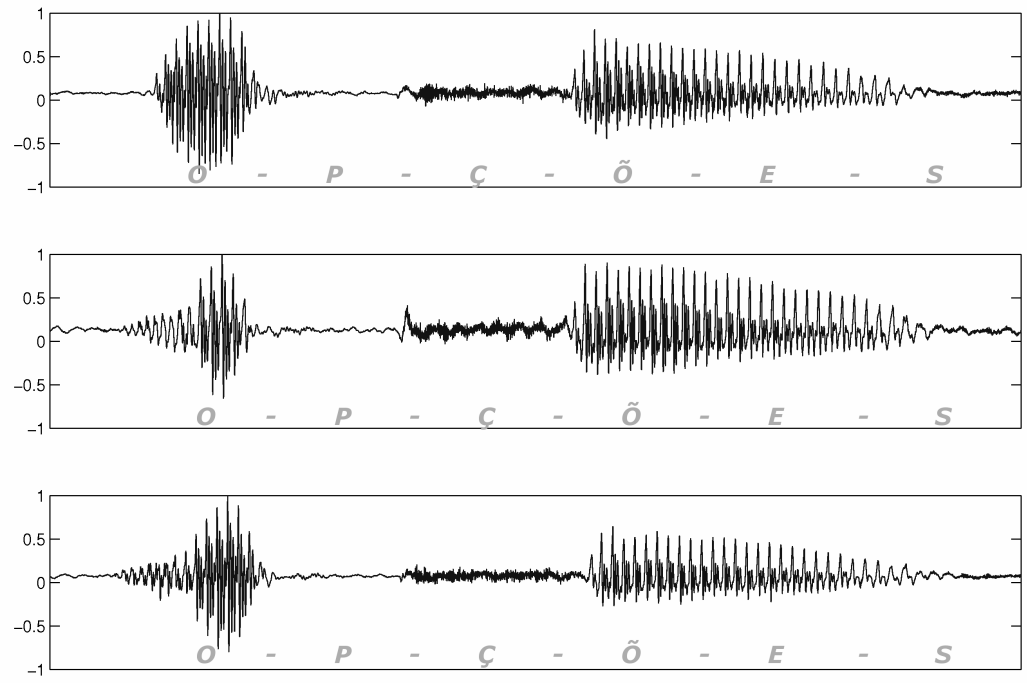



Figure 2.1: Three utterances from the same speaker for the word OPÇÕES /op'sõyʒ/.

### 2.2.1 Human Sound Production System Based Models

#### Linear predictive coding

Linear prediction theory can be traced back to the 1941 work of Kolmogorov[1] referenced in Vaidyanathan [68]. In this work, Kolmogorov considered the problem of extrapolation of discrete time random processes. From the first works that explored the application in speech coding, we can dettach the work of Atal and Schroeder [3] and that of Itakura and Saito[2] referenced in Vaidyanathan [68]. Atal and Itakura independently formulated the fundamental concepts of Linear Predictive Coding (LPC).

Itakura [27], Rabiner and Levinson [54], and others, have started the proposition of using LPC with pattern recognition technologies to enable speech recognition applications. LPC represents the spectral envelope of a speech digital signal, using the information of a linear predictive model. LPC is actually a close approximation of a speech production system: the speech is produced by a buzzer (glottis), characterized by loudness and pitch, at the end of a tube (vocal tract), characterized by its resonance, with occasional hissing and popping sounds (made by tongue, lips and throat).

LPC is widely used in speech analysis and synthesis. Some recent examples of speech recognition systems that uses LPC could be the work of Thiang [66], that uses LPC to extract word data from a speech signal in order to control the movement of a mobile robot, and the work of Paul [51], that presents the Bangla speech recognition system which uses LPC and cepstral coefficients to construct the codebook for the artificial neural network.

[1]Kolmogorov, A. N. Interpolation and extrapolation of stationary random sequences, *Izv. Akad. Nauk SSSR Ser. Mat.* 5, pp. 3–14, 1941.

[2]Itakura, F., and Saito, S. A statistical method for estimation of speech spectral density and formant frequencies, *Trans IECE Jpn.*, vol. 53-A, pp. 36–43, 1970.

### Cepstrum

The term *cepstrum* was first coined by Bogert et al.[3], referenced in Oppenheim and Schafer [48], and they mean "the spectrum of the log of the spectrum of a time waveform". The spectrum of the log spectrum shows a peak when the original time waveform contains an echo. This new spectral representation domain is not the frequency nor the time domain. Bogert et al. chose to refer to it as the *quefrency* domain.

The cepstrum could be very important to many speech recognition systems. As stated by Oppenheim and Schafer [48], "[...] the cepstral coefficients have been found empirically to be a more robust, reliable feature set for speech recognition and speaker identification than linear predictive coding (LPC) coefficients or other equivalent parameter sets". There are a vast family of cepstrum, several of them applied with success to speech recognition systems, like linear prediction cepstrum [70], shifted delta cepstrum[4] [8], and mel-frequency cepstrum [16].

An example on using cepstrum to speech recognition could be found in the work of Kim and Rose [32] that proposes a cepstrum-domain model combination method for automatic speech recognition in noisy environments. As a recent example, we can dettach the work of Zhang et al. [72] that uses time-frequency cepstrum, based on a horizontal discrete cosine transform of the cepstrum matrix for de-correlation, and performs a heteroscedastic linear discriminant analysis to achieve a novel algorithm to language recognition field.

[3]B.P. Bogert, M.J.R. Healy, and J.W. Tukey, "The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking, in *Time Series Analysis*, M. Rosenblatt, Ed., 1963, ch.15, pp. 209243.

[4]Proposed in the work of B. Bielefeld, "Language identification using shifted delta cepstrum," In *Fourteenth Annual Speech Research Symposium*, 1994. Referenced in Carrasquillo et al.

### 2.2.2 Human Auditory System Based Models

Inside feature extraction theme, the human auditory system model is discussed for over a hundred years. Actually, the search for a reliable auditory filter to extract the best parametric representation of acoustic data is the main quest for several researchers worldwide. The auditory filters, according to Lyon et al. [38], "include both those motivated by psychoacoustic experiments, such as detection of tones in noise maskers, as well as those motivated by reproducing the observed mechanical response of the basilar membrane or neural response of the auditory nerve". They have also stated that "today, we are able to represent a wide range of linear and nonlinear aspects of the psychophysics and physiology of hearing with a rather simple and elegant set of circuits or computations that have a clear connection to underlying hydrodynamics and with parameters calibrated to human performance data".

In this section, we present some of those human auditory system models that enables the extraction of different features from the speech signals.

#### Mel-frequency Cepstrum

From the family of cepstrum, the Mel-frequency Cepstrum is widely used for the excellent recognition performance they can provide [16]. Mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. The mel scale (comes from "melody" scale), proposed by Stevens et al. [64], is based on pitch comparisons from listeners, i.e., mel scale is a perceptual scale of pitches that were judged equal in distance from one another by tested listeners. Davis and Mermelstein [16]

is widely referenced when exploring Mel-frequency Cepstrum history, but Mermelstein usually credits John Bridle's work[5] for the idea.

According to Combrinck [9], "from a perceptual point of view, the mel-scaled cepstrum takes into account the non-linear nature of pitch perception (the mel scale) as well as loudness perception (the log operation). It also models critical bandwidth as far as differential pitch sensitivity is concerned (the mel scale)". Widely used in speech recognition systems, we can found a recent example in the work of Bai and Zhang [4], where they extract linear predictive mel cepstrum features through "the integration of Mel frequency and linear predictive cepstrum". Another recent example, the work of Kurian and Balakrishnan [34], extracts MFC coefficients to implement a speech recognition system for the recognition of Malayalam numbers.

### Gammatone Filterbank

Aertsen and Johannesma [2] had introduced the concept of gammatone, "an approximative formal description of a single sound element from the vocalizations [...]". According to Patterson [50], the response of a gammatone filter "transduces" the basilar membrane motion, converting it into a multi-channel representation of "[...] the pattern of neural activity that flows from the cochlea up the auditory nerve to the cochlear nucleus". In other words, Patterson[6], referenced in [63], states that "Gammatone filters are derived from psychophysical and physiological observations of the auditory periphery and this filterbank is a standard model of cochlear filtering".

[5] J. S. Bridle and M. D. Brown, "An experimental automatic word recognition system," Tech. Rep. JSRU Report No. 1003, Joint Speech Research Unit, Ruislip, England, 1974.

[6] R. D. Patterson, *et al.*, "Auditory models as preprocessors for speech recognition," in *The auditory processing of speech: From sounds to words*, M.E.H. Schouten, Ed., Berlin, Germany: Mouton de Gruyter, pp. 67-83, 1992.

A gammatone filterbank is composed of as many filters as the desired number of output channels. Each channel is designed to have a center frequency and a respective equivalent rectangular bandwidth (ERB). The filter center frequencies are distributed across frequency in proportion to their bandwidth. Greenwood had come with the assumption that critical bandwidth represent equal distances on the basilar membrane and also had defined a frequency-position function [22]. Glasberg and Moore [21] have summarized human data on the ERB for an auditory filter.

The preference of using gammatone filters to simulate the cochlea is sensed by Schluter et al. [60], "the gammatone filter (GTF) has been hugely popular, mostly due to its simple description in the time domain as a gamma-distribution envelope times a tone". Recent examples of using GTF includes the works of Shao and Wang [63], and Shao et al. [62], that respectively proposes and applies the gammatone frequency cepstrum, based on the discrete cosine transform applied to features extracted from the gammatone filter response to speech signal. Another example is the work of Schluter et al. [60] that presents "an acoustic feature extraction based on an auditory filterbank realized by Gammatone filters".

### Wavelet filterbank

The name wavelet was firstly coined by Morlet et al.[7]. The wavelet transform is described by Daubechies [14] as "a tool that cuts up data or functions or operators into different frequency components, and then studies each component with a resolution matched to its scale". Daubechies also made the correspondence between the human auditory system and the wavelet transform, at least when analyzing the basilar membrane response to the pressure

[7]J. Morlet, G. Arens, I. Fourgeau, and D. Giard, "Wave propagation and sampling theory", *Geophysics*, 47, pp. 203-236, 1982. Referenced in [14].

amplitude oscillations transmitted from the eardrum. According to her, "the occurrence of the wavelet transform in the first stage of our own biological acoustical analysis suggests that wavelet-based methods for acoustical analysis have a better chance than other methods to lead, e.g., to compression schemes undetectable by our ear".

The work of Gandhiraj and Sathidevi [19] proposes a cochlea model based on a high resolution Wavelet-Packet filterbank. The latest published works from this research [52, 53] have used Wavelet-Packet filterbank to extract features from the speech signals. Another example of recent works using wavelets for speech recognition is the work of Maorui et al. [40], that uses the wavelet packet transform to evaluate what they present as the improved mel-frequency cepstral coefficients.

### Two Filter Cascades

As stated by Lyon et al. [38], the polezero filter cascade (PZFC) "has a much more realistic response in both time and frequency domains, due to its closer similarity to the underlying wave mechanics, and is not much more complicated". Since [37], Lyon has stated that "the filter-cascade structure for an cochlea model inherits two key advantages from its neuromorphic roots: efficiency of implementation, and potential realism".

We have not found recent works that use this approach as the parametric representation of speech, but we realize that PZFC is worth to be investigated.

## 2.3 Speech Recognition

Speech recognition is considered a pattern recognition problem, where each kind of phoneme, spoken word, emotion, or even the individual identity of a given set of speakers constitutes a different class to try fitting my testing pattern. This pattern is constituted by the features which are extracted from each speech signal analyzed by the recognizer system. A successful recognition means the recognizer system successfully indicates the expected class of a given input speech signal. Some of the most widely used recognizers are presented in the following. As convention, training set identifies the set of patterns presented to the recognizer in the learning stage; testing set identifies the set of patterns, or an individual pattern, that the recognizer must make an inference about which class it belongs, after the learning stage.

### Clustering Algorithms

From all various techniques that can be considered a clustering algorithm, we shall consider one elementary, named *K-means* clustering. We start with the evaluation of each predetermined class mean, i.e., the mean of all n-dimensional pattern vector obtained by the feature extraction stage for all speech signals that we know from that specific class. Those means are known as class centroids. K-means clustering algorithm will check for each pattern vector of the testing set which centroid is the one most closer to it. This measurement of distance can vary from an implementation to another. Common used distance measurements include the widely known Euclidean distance, Mahalanobis distance[8], or even Itakura-Saito distance [27].

[8]P.C. Mahalanobis, "On the generalised distance in statistics," *Proceedings of the National Institute of Science of India* 12, pp.49-55, 1936; referenced in [39].

### Support Vector Machine

Cortes and Vapnik [11] have proposed the Support-vector Networks, later named as Support Vector Machines (SVM). They have presented SVM as "a new learning machine for two-group classification problems". Basically, SVM relies on the preprocessing of the data to represent patterns in a much high dimension. The main idea is that when applying an appropriate nonlinear mapping function in order to augment the original feature space to a sufficiently high dimension, data from two categories can always be separated by a hyperplane.

From the recent works mentioned before, the works [4, 72] uses SVM as the speech recognition system. For more information about SVM, we suggest the work of Burges [7].

### Artificial Neural Networks

According to Fausett [17], an Artificial Neural Network (ANN) is "an information-processing system that has certain performance characteristics in common with biological neural networks". The motivation of studying ANNs is their similarity to successfully working biological systems, which consist of very simple but numerous nerve cells (neurons) that work massively parallel and have the capability to learn from training examples [33]. The main consequence of this learning ability of ANNs is that they are able of generalizing and associating data, i.e., ANN can make correct inferences about data that were not included in the training examples.

The first attempt to simulate a biological neuron was performed in 1943 by McCulloch and Pitts [43]. Since then, several neuron approaches and architectures have been developed increasing this way the family of artificial neural networks. One of the most widely used ANN architectures for speech

recognition is the Multilayer Perceptron (MLP), or multilayer feed-forward networks [59], based on Rosenblatt's Perceptron neuron model [58].

Note that the universal approximation theorem from mathematics, also known as the Cybenko theorem, claims that the standard multilayer feed-forward networks with a single hidden layer that contains finite number of hidden neurons, and with arbitrary activation function, are universal approximators on a compact subset of $\Re^n$ [12]. This theorem was first proved by Cybenko[13] for a sigmoid activation function. Hornik [25] concludes that "it is not the specific choice of the activation function, but rather the *multilayer feedforward architecture* itself which gives neural networks the potential of being universal learning machines" [emphasis in original].

From the recent works mentioned before, the works [19, 51, 52, 53, 40] use ANN architectures as the speech recognition system. For more information about ANN, we suggest the works of Fausett [17], Haykin [23], and Kriesel [33].

### Hidden Markov Models

The Hidden Markov Model (HMM) can be defined as a *finite set of states*, each of which is associated with a probability distribution, multidimensional in general. There are a set of probabilities called *transition probabilities* that rules over the transitions among the states. According to the associated probability distribution, a particular state can generate one of the possible outcomes. The system being modeled by a HMM is assumed to be a Markov process with unobserved (or hiddden) states. A Markov process is defined by a sequence of possibly dependent random variables with the property that any prediction of the next state of the sequence may be based only on the

last state. In other words, the future value of such a variable is independent of its past history.

From the recent works mentioned before, the works [32, 66, 34] use HMM as the speech recognition system. For more information about HMM, we suggest the work of Rabiner [55].

### Hybrid HMM/ANN

Hybrid HMM/ANN systems combine artificial neural networks (ANN) and hidden Markov models (HMM) to perform dynamic pattern recognition tasks. In the speech recognition field, hybrid HMM/ANN can lead to very powerful and efficient systems, due to the combination of the discriminative ANN capabilities and the superior dynamic time warping HMM abilities [57]. One of the most popular hybrid approach is described by Hochberg et al. [24]. Rigoll and Neukirchen [57] have presented a new approach to hybrid HMM/ANN which performs, as stated by them, "already as well or slightly better as the best conventional HMM systems with continuous parameters, and is still perfectible".

The hybrid HMM/ANN systems are a modified form of an earlier design known as "discriminant HMMs" which was initially developed to directly estimate and train ANN parameters to optimize global posterior probabilities. In hybrid HMM/ANN systems, all emission probabilities can be estimated to the ANN outputs and those probabilities are referred to as conditional transition probabilities [6].

Recent works using HMM/ANN hybrid models include the work of [31], which proposes a novel enhanced phone posteriors to improve speech recognition systems performance (including HMM/ANN), and the work of Bo et al. [5], which applied HMM/ANN in order to improve the performance of a

existing speech access control system. Another example, the work of Huda et al. [26], based on the extraction of distinctive phonetic features, proposes the use of two ANN stages with different purposes before reaching a HMM-based classifier.

For more information about hybrid HMM/ANN, we suggest the work of Bourlard and Morgan [6].

## 2.4 Choices for this Work

From earlier conducted tests based on this project database, we have chosen *Mel-frequency Cepstrum* as the parametric representation of our acoustic data. Because of its simple implementation and great ability of generalization, we have chosen an Artificial Neural Network architecture, the *Multilayer Perceptron*, as the recognizer system for this project. Details on those can be found in Chapter 3.

For the purposes of this work, we have chosen *Classical* and *Bottom-up* methods to compare performances with this project proposed word boundary detector based on the Teager energy operator. The proposed word boundary detector method and the comparisons with those other methods can be found in Chapter 4.

# Chapter 3

# Theoretical Background

The results derived from this work may support the implementation of speech recognition systems described by the parameters shown in Table 3.1. Those parameters are based on the ones proposed by Zue, Cole and Ward [75].

Table 3.1: Proposed speech recognition system

| **Parameters** | **Range** |
|---|---|
| Speaking Mode | Isolated words |
| Speaking Style | Spontaneous speech |
| Enrollment | Speaker-independent |
| Vocabulary | 50 words |
| Language Model | Finite-state |
| SNR | High ($\approx$ 30dB) |
| Transducer | Electret condenser microphone |

Regarding high Signal to Noise Ratio (SNR) here stated, the system described in this work could also act in lower SNR environments, but some configuration adjustments are required in the speech segmentation stage described in Chapter 4.

The recognition system described in the present work is build with widely used steps, but the simpler ones. Besides speech segmentation, fully developed inside this research, all other stages could be found in related literature. The proposed system block diagram for this research project is presented in figure 3.1. Each block is explained throughout this chapter.

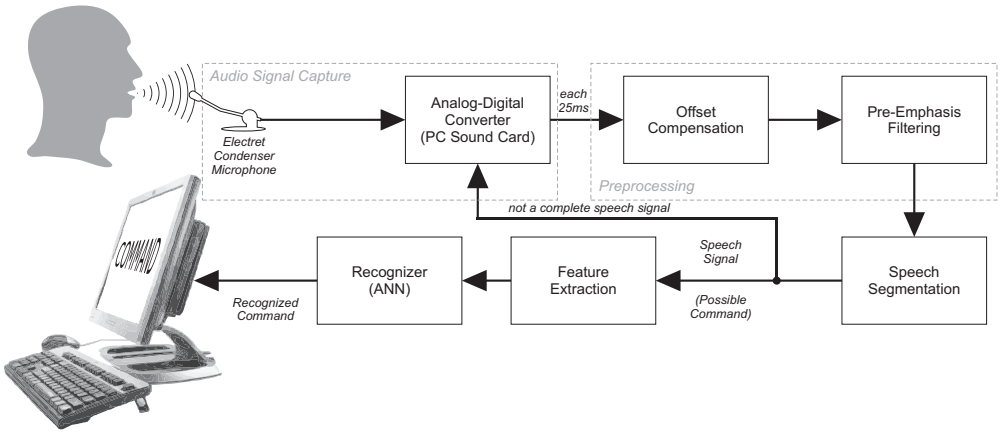



Figure 3.1: Proposed speech recognition system block diagram.

## 3.1 Audio Signal Capture

By using a electret condenser microphone, the audio signal is captured using the sampling frequency of 8kHz and the word-length of 16 bits. As stated by Nyquist-Shannon sampling theorem[61], "if a function $f(t)$ contains no frequencies higher than $W$ cps[1], it is completely determined by giving its ordinates at a series of points spaced $\frac{1}{2W}$ seconds apart". So, capturing an audio signal with a sampling frequency of 8kHz, one can ensure the reliable capture of only 0 to 4kHz frequency components from the given signal.

[1]Character per second (cps) is a unit of data signaling rate (DSR) that express the number of characters passing a designated point per second. In signal processing area, we can understand it as Hertz (Hz).

The specifics of the analog-to-digital conversion are not part of the present dissertation. Applications which will be developed in this research will use the analog-to-digital converter featured in most PC sound cards.

Meanwhile, for this work, a database of prerecorded audio files (see Section 1.3) was used to evaluate the potentialities of the system. For the conducted tests shown in this work, the loaded recorded audio files were constituted by values inside the [-1,1] interval.

## 3.2 Preprocessing

European Telecommunications Standards Institute has published the ES 201 108 V1.1.3[1] standard[2] to guide transmissions over mobile channels. ETSI Standards are respected over all in Europe, including Portugal, which ensures ETSI concerns about Portuguese language among others. ES 201 108 standard has significant information on preprocessing for speech signals.

### 3.2.1 Offset compensation

After analog-to-digital conversion, a notch filtering operation is applied to the digital samples of the input audio signal to remove their DC offset, producing a offset-free input signal. This operation is also known as zeropadding. The notch filtering operation is done as presented in equation (3.1), the same as in item (4.2.3) from ES 201 108 standard [1]:

$$s_{of}(n) = s_{in}(n) - s_{in}(n-1) + 0.999 \cdot s_{of}(n-1), \tag{3.1}$$

[2]ETSI Standard for Speech Processing, Transmission and Quality aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms.

where $s_{of}$ is the offset-free input signal; $s_{in}$ is the input signal; and $n$ is the sample index. It assumes, for $n = 0$, that $s_{of}(0) = s_{in}(0)$.

### 3.2.2 Pre-emphasis filtering

Before any other computation, the offset-free audio signal is then processed by a simple pre-emphasis filter. After pre-emphasis filtering, the average speech spectrum turns to roughly flat.

The application of the pre-emphasis filter to the offset-free input signal is done by applying Equation (3.2), the same as in item 4.2.6 from ES 201 108 standard:

$$s_{pe}(n) = s_{of}(n) - 0.97 \cdot s_{of}(n-1), \tag{3.2}$$

where $s_{pe}$ is the pre-emphasis filtered input signal; $s_{of}$ is the offset-free input signal; and $n$ is the sample index. It assumes, for $n = 0$, that $s_{pe}(0) = s_{of}(0)$.

## 3.3 Speech Segmentation

The spoken word segmentation algorithm pretends to detect the starting and the ending points of a speech waveform within a sampled audio waveform signal. This process is known as boundary detection of the speech, detection of the speech endpoints, speech segmentation, or spoken word isolation.

A novel method for speech segmentation, based on the Teager Energy Operator (TEO), was developed during this research. This method is detailed in Chapter 4, named as "TEO-based method for Spoken Word Segmentation" (TSWS). The result of this stage is a word-like speech signal segmented from original recorded signal.

### 3.3.1 The Teager Energy Operator

In the work of Teager and Teager[3] on nonlinear modeling of speech, referenced in Maragos et al. [41], an energy operator on speech-related signals is first presented.

In his work, Kaiser has discussed the properties of that Teager's energy-related algorithm — later designed as the Teager Energy Operator (TEO), or the Teager-Kaiser Operator — which, "by operating on-the-fly on signals composed of a single time-varying frequency, is able to extract a measure of the energy of the mechanical process that generated this signal" [29].

When the signal consists of several different frequency components — like in captured speech signals — Kaiser states that, to use this energy operator effectively, "it is important to pass the signal through a bank of bandpass filters first; the algorithm is then applied to the outputs from each of these bandpass filters" [29].

Kaiser [30] has also defined both TEO in the continuous and discrete domains as "very useful 'tools' for analyzing single component signals from an energy point-of-view" [emphasis in original].

TEO is then defined by Equation (3.3), in the continuous domain, and by Equation (3.4), in the discrete domain [30]. Note that, in the discrete domain, this algorithm uses only three arithmetic operators applied to three adjacent samples of the signal for each time shift.

$$\Psi\left[x(t)\right] \triangleq \left(\frac{\mathrm{d}x(t)}{\mathrm{d}t}\right)^2 - x(t) \cdot \frac{\mathrm{d}^2x(t)}{\mathrm{d}t^2}, \tag{3.3}$$

where $\Psi$ is the TEO operator; and $x(t)$ is the amplitude of the signal at the time $t$.

[3] H. M. Teager and S. M. Teager, "Evidence for Nonlinear Production Mechanisms in the Vocal Tract," *NATO Advanced Study Institute on Speech Production and Speech Modeling*, Bonas, France, July 1989; Kluwer Acad. Publ., Boston, MA, 1990.

$$\Psi\left[x(n)\right] = x_n^2 - x_{n-1} \cdot x_{n+1}, \tag{3.4}$$

where $\Psi$ is the TEO operator; and $x(n)$ is the $n^{th}$ sample of the discrete signal.

Another aspect of TEO, observed when it is applied to speech signals without a bandpass filterbank previous stage, is interesting for this research: when one applies TEO to signals composed of two or more frequency components — a speech signal, for example — TEO does not give the energy of the system generating this composite signal, but, according to Kaiser [29], "it is as if the algorithm is able to extract the envelope function of the signal".

## 3.4 Feature Extraction

According to Martins [42], to evaluate the features (or coefficients) to be extracted from the input speech signals, it is usual to divide those signal into non-overlapping frames. Each of those frames has to be multiplied with a Hamming window, presented in equation (3.5), in order to keep the continuity of the first and the last points in the frame [16], [1]. To ensure an overlapping relation between windows and frames, each window size must be greater than the frame size, as shown in Figure 3.2.

$$h(n) = \begin{cases} 0.54 - 0.46 \cdot \cos\left(\frac{2\pi n}{N-1}\right), & \text{if } 0 \leq n \leq N-1 \\ 0, & \text{otherwise} \end{cases}, \tag{3.5}$$

where $h(n)$ is the window resultant of $n^{th}$ sample of the frame; and $N$ is the total number of samples from each frame.

Martins has also designed a method to keep the same numbers of coefficients extracted from each signal. In this method, each segmented speech

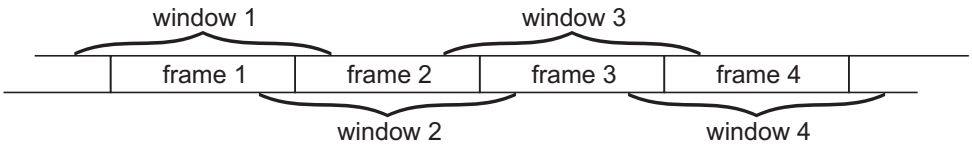



Figure 3.2: Overlapping of windows on frames for coefficient evaluation.

signal is divided into a fixed number of 80 frames. A window of 20ms [4] is then chosen to run through the frames. If the evaluated frame size is greater than the window size, turning the window overlap impossible, the window size is then adjusted to 1.5 times the frame size. The algorithm for setting frame and windows sizes, evaluated in number of samples, is presented in Algorithm 3.1.

**Algorithm 3.1** Setting frame and window sizes.

$fr \leftarrow \text{length}(signal)/80$ // frame size ($fr$)
$wd \leftarrow 0.02 * Fs$ // sampling frequency ($Fs$), window size ($wd$)
**if** $fr > wd$ **then**
    $wd = 1.5 * fr$
**end if**

Different timbres in speech signals correspond to different energy distribution over frequencies, as can be shown by a spectral analysis. Therefore, the Fast Fourier Transform (FFT) is performed to obtain the magnitude frequency response of each "windowed" frame.

After, the magnitude frequency response obtained by FFT is multiplied by a set of 16 triangular bandpass filters, in order to get the log-energy of each filter respective output. The center frequencies of those filters are equally spaced along the Mel frequency scale. The Mel scale (comes from "melody" scale), proposed by Stevens, Volkman and Newman[64], is based on pitch comparisons from listeners, i.e., Mel scale is a perceptual scale of pitches that were judged equal in distance from one another by tested listeners. The

[4] For a frequency sample of 8kHz, 20ms means a windows size equal to 160 samples.

relation from Hertz to Mels is achieved by the equation (3.6), as presented by O'Shaughnessy [49]. The relation plot is shown in Figure 3.3. Table 3.2 and Figure 3.4 presents the center frequencies and the bandwidth of each of the 16 triangular bandpass filters.

$$m = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right), \tag{3.6}$$

where $m$ is the frequency in Mels; and $f$ is the frequency in Hz.

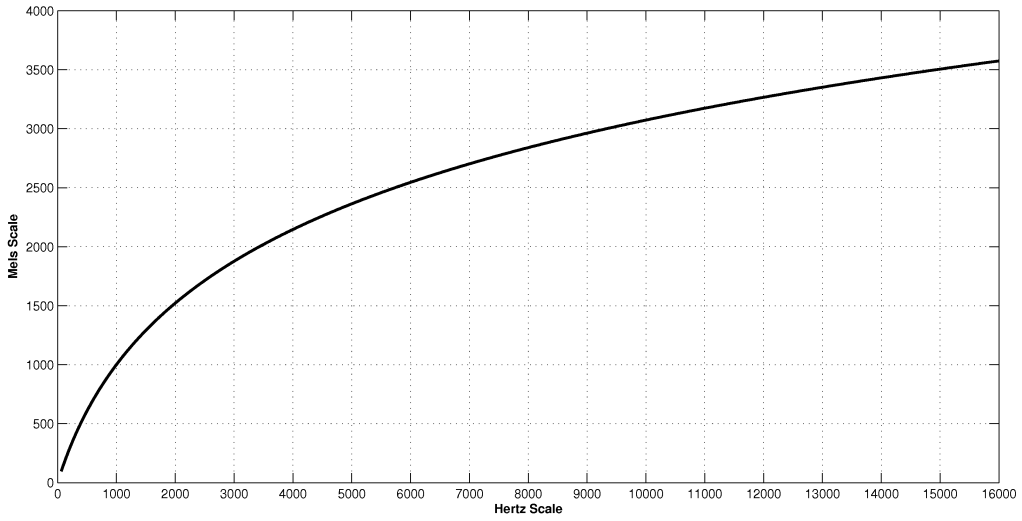



Figure 3.3: Hertz scale versus Mel scale.

Finally, Mel-frequency Cepstral Coefficients (MFCC) could be extracted from the input speech signal. The choice of using MFCC as features to be extracted from speech signals comes from the widely use of those coefficients and the excellent recognition performance they can provide [16, 42]. MFCC, generalized from the ones computed by Davis and Mermelstein [16], is presented in Equation (3.7).

Table 3.2: Center frequencies and respective bandwidth for the designed 16 triangular bandpass filters.

| **Channel** | **Center Frequency [Hz]** | **Bandwidth [Hz]** |
|---|---|---|
| 1 | 83 | 176 |
| 2 | 176 | 197 |
| 3 | 280 | 220 |
| 4 | 396 | 246 |
| 5 | 526 | 275 |
| 6 | 671 | 308 |
| 7 | 833 | 344 |
| 8 | 1,015 | 385 |
| 9 | 1,218 | 431 |
| 10 | 1,446 | 482 |
| 11 | 1,700 | 539 |
| 12 | 1,984 | 603 |
| 13 | 2,303 | 674 |
| 14 | 2,659 | 754 |
| 15 | 3,057 | 843 |
| 16 | 3,502 | 943 |

$$\mathrm{MFCC}_i = \sum_{k=1}^{N} X_k \cdot \cos\left[ i \left( k - \frac{1}{2} \right) \frac{\pi}{N} \right], \quad i = 1, 2, \ldots, M, \tag{3.7}$$

where $M$ is the number of cepstral coefficients; $N$ is the number of triangular bandpass filters; and $X_k$ represents the log-energy output of the $k^{th}$ filter.

The set of MFCC constitutes the Mel-frequency Cepstrum (MFC), which is derived from a type of cepstral representation of the audio signal. The main difference from a normal cepstrum[5] is that MFC uses frequency bands equally spaced on the Mel scale (an approximation to the response of human auditory system) and normal cepstrum uses linearly-spaced frequency bands.

[5]A cepstrum is termed as "the spectrum of the log of the spectrum of a time waveform" [48].

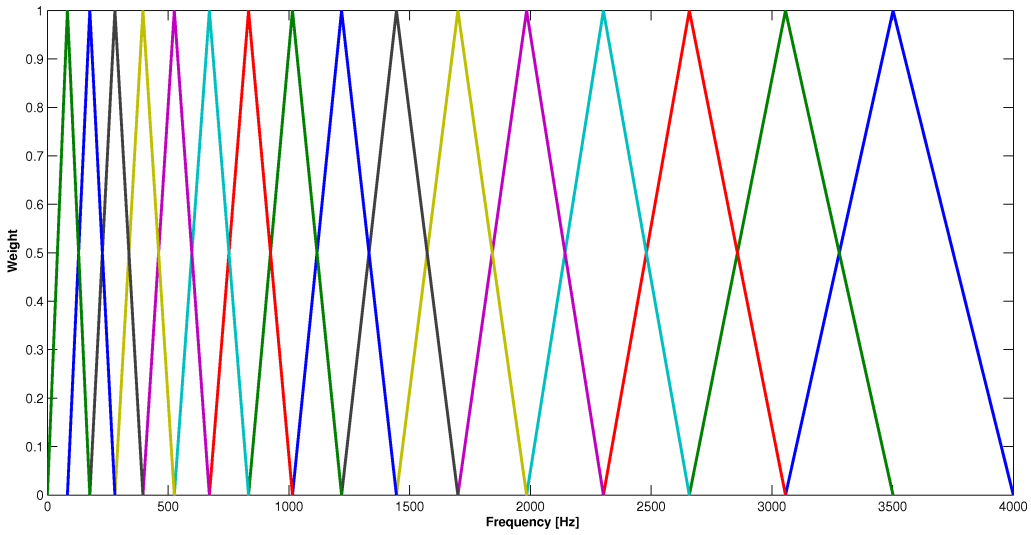



Figure 3.4: Filters for generating Mel-Frequency Cepstrum Coefficients.

For a given input speech signal, we start dividing it into 80 frames and, for each frame, we end up evaluating 16 MFCC. Concatenating all those coefficients in a row, we get 1,280 coefficients (the feature vector) to act as the input vector for the recognizer system.

## 3.5 The Recognizer

The option for using an artificial neural network (ANN) model to act as the recognition system is just an aspect of this work. As stated before, an *artificial neural network* is defined by Fausett as "an information-processing system that has certain performance characteristics in common with biological neural networks" [17]. Fausett also characterizes an ANN by its architecture, its activation function, and its training (or learning) algorithm. The breadth of ANN's applicability is suggested by the areas in which they are currently being applied: signal processing, control, pattern recognition, medicine, business, among others. Speech recognition is also an area where

ANNs are being applied with great success rates [65]. However, some hybrid model solutions, as the ones which combine ANN with hidden Markov models (HMMs), have shown better results for speech recognition systems [5, 26, 31].

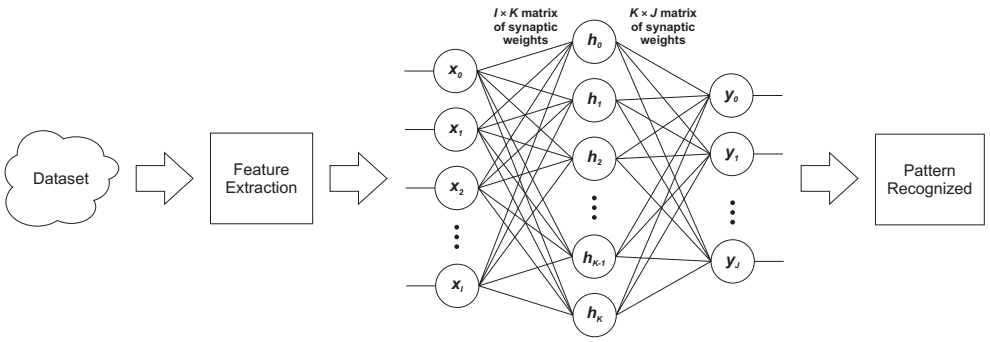



Figure 3.5: Diagram with a $I \times K \times J$ MLP acting as recognizer.

Multi-layer Perceptron (MLP) is an ANN architecture relatively simple to implement. It is very robust when recognizing different patterns from the ones used for training, and it has wide spread use to handle pattern recognition problems. Based on Rosenblatt's Perceptron neuron model [58], it typically uses an approach based on the Widrow-Hoff backpropagation of the error [17] as the supervised learning method. Figure 3.5 presents the diagram of a general MLP with $I$ input units, $K$ hidden units, and $J$ output units. Note that the number of input units from a MLP is the same number of coefficients generated after the feature extraction stage. Likewise, the number of output units is generally the number of existing classes for pattern classification (or recognition).

The desire of effectively recognize isolated spoken words can be classified as a pattern recognition (or speech recognition) problem. Thereafter, because

of robustness and simplicity of implementation, a single hidden layer MLP[6] is the chosen ANN architecture for this project recognizer.

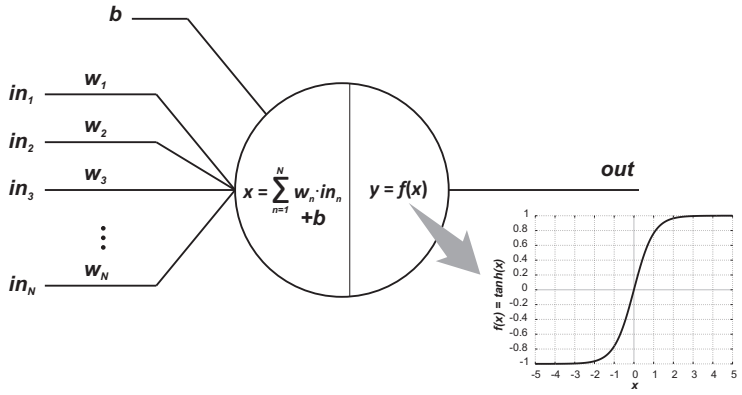



Figure 3.6: A Perceptron with the hyperbolic tangent as the activation function.

Each neuron of the chosen MLP, presented in Figure 3.6, has the hyperbolic tangent (*tanh*) as the activation function. Neurons work as mathematical functors, using as argument the weighted arithmetic mean from input values, weighted by their respective synaptic weights ($w_i$) plus a bias ($b$). The output of a neuron is the result of applying the activation function in that weighted mean argument. The neuron used in the ANN architecture could be described by equation (3.8). Activation of a neuron means that the result of that neuron's function (output level) is above a predetermined threshold.

$$O_m = \tanh\left(b_m + \sum_{n=1}^{N} w_{n,m} \cdot I_n\right), \tag{3.8}$$

where $O_m$ is the expected output of the $m^{th}$ neuron; $b_m$ is the bias value for the $m^{th}$ neuron; $w_{n,m}$ is the synaptic weight of the $m^{th}$ neuron correspondent

[6]The literature is emphatic when stating that, for nearly all problems, one hidden layer for a MLP is enough. Some works found out that two hidden layers should be implemented for modeling data with discontinuities, and there is no theoretical reason for using more than two hidden layers.

to $n^{th}$ input; $N$ is the total number of inputs; and $I_n$ is the $n^{th}$ input of the neuron.

The concept of training an ANN means iterative updates of the ANN's synaptic weights and biases. Supervised training means the training algorithm takes into account the difference of the expected output vector (or target vector) and the actual output vector generated when ANN is exposed to a single input vector. This accounting is part of the correction term calculation for the synaptic weights updates. The training process is repeated until the ANN reaches an acceptable total error of performance, or it achieves a maximum number of training epochs.

Different types of algorithms were verified to enable a fast training for the recognizer. *Scaled Conjugate Gradient* [46] was then chosen as the supervised training algorithm and *Bayesian Regularization* algorithm[7] [67] was chosen to enable ANN's performance evaluation during the training.

The project recognizer is then implemented as a single hidden layer MLP with $1,280$ input units, 100 hidden units and 50 output neurons. Each output neuron corresponds to a different voice command and its activation is equivalent to the recognition of its respective command (see Table 1.2 for the group of voice commands to be recognized). The project MLP uses real numbers as input vectors (input $\in \Re$) and bipolar target output vectors (target output could be -1 or 1).

### 3.5.1 Confusion Matrices

Confusion matrix (CM), or table of confusion, is a visualization tool typically used in supervised learning pattern recognition systems. Each row of the matrix represents the actual output class recognized by the system, while

[7] *Bayesian Regularization* is also known as *Mean Squared Error with Regularization*

each column represents the correct target class the system should recognize. Confusion matrices enable an easy visualization of the system mislabels, i.e., the system confusion rates on the decision about which class to recognize.

# Chapter 4

# Proposed Method for Speech Segmentation

As stated by Lamier, Rabiner, et al [35], "accurate location of the endpoints of an isolated word is important for reliable and robust word recognition". Due to its importance, we had searched for a reliable and robust speech segmentation method. Finally, during this research, we have developed a novel method for speech segmentation, based on the Teager Energy Operator (TEO), also known as the Teager-Kaiser Operator. The proposed method was named "TEO-based method for Spoken Word Segmentation" (TSWS). The TSWS method has evolved from the premise that TEO can emphasize speech regions from an audio signal, as presented in other works [53].

The following chapter presents TSWS as a method for speech segmentation, the results achieved when applying it to an American English support database, and comparisons with Classical [56] and Bottom-up [35] speech

segmentation methods. Note that preprocessing stage present in Figure 3.1 (block diagram) is already included in the TSWS algorithm.

## 4.1 Support Database

In order to explore all potentialities from the TSWS method, we choose to work with another database, named support database. The chosen one was kindly provided by Compton[1] [10] and it aims to represent nearly all phonetic sounds of American English. This database is composed by 68 audio recordings from the same female speaker, each of the recordings containing from 3 to 6 words besides the sound of the phoneme itself. These audio samples were recorded in a studio and present low noise level (SNR$\gg 30dB$, clear signal). For this research, all files from the support database were converted from *MPEG-2 Audio Layer 3* format (*mp3* file extension) to *WAVEform Audio* format (*wav* file extension). Those files were also converted to the sampling frequency of 8kHz and the word-length of 16 bits. Additionally, all audio files were divided between the words they contain and their respective phoneme sounds were discarded. The support database ends up with 258 audio signals. A complete list of achieved words can be found in Appendix A.

There are two main reasons for choosing this support database. First, we could check the behavior of the TSWS method when it faces high diversified phonetic sounds of American English. Note that *stops*, *frictions*, *glides* and *nasals*, which are hard types of phonemes to segmentation methods due to low energy presented, are also included in this database. Second, no similar Brazilian Portuguese database has been found. There is no guarantee that

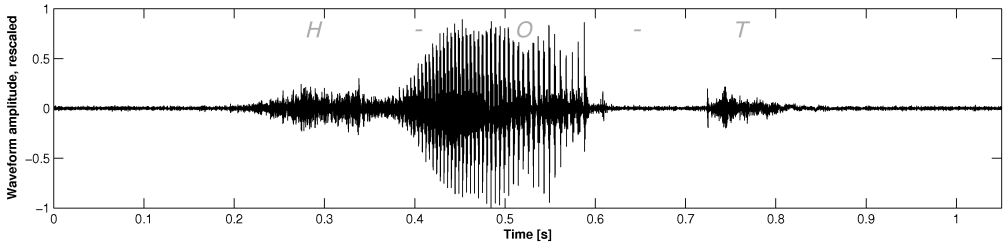
[1]Arthur J. Compton, PhD. Institute of Language and Phonology, Compton Phonological Assessment of Foreign Accent.

the database used in this project include all diversified phonetic sounds from Brazilian Portuguese.

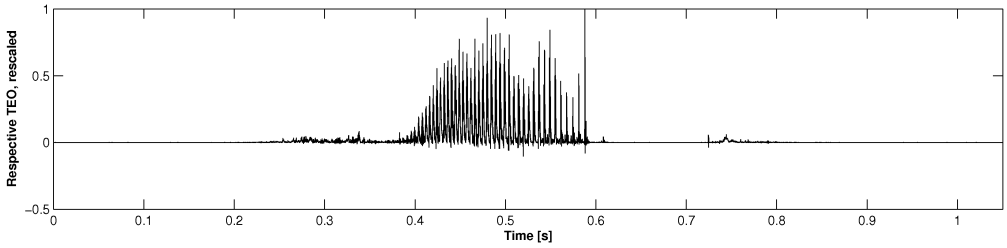
## 4.2 Proposed TEO-Based Segmentation

The development of the TSWS method has started with the awareness that TEO can give emphasis to speech regions in audio waveforms at the same time it understates the noise-only regions. One aspect of the conclusions on the work of Kaiser [29] states that TEO seems to be able to extract the envelope function of a signal composed of two or more frequency components when not working in the output of a bandpass filterbank. This aspect give us the indication that the awareness into we are basing TSWS development is reliable. Figure 4.1 presents a given original audio waveform and the respective TEO resultant waveform. Note that both waveforms were divided by their respective maximum absolute values for visualization purposes only. It can be seen in Figure 4.1(b) that, from an 'energy point-of-view', the speech carries much more information than the noise captured from the environment.

Application of the TSWS method to constantly incoming audio signals, instead of complete recorded audio signals, requires the use of a non-overlapping frame-by-frame approach. A non-overlapping frame of 25ms is then set ($T_{frame}$) to be the elementary structural constituent of the captured input audio, as suggested by item 4.2.4 from ES 201 108 standard [1]. The first captured frames, for the length of time chosen previously ($T_{silence}$), are identified as "silence". At this moment, the TSWS method constructs a vector formed by TEO values of non-speech samples, named as silence vector. Now, there is the need for setting a reference value to the decision if the subsequent




(a) Original audio waveform.




(b) Respective TEO resultant waveform.

Figure 4.1: Audio waveform for the English word "HOT" /hat/ with white noise addition (SNR 30dB).

captured frames includes speech information or not. That reference value is evaluated by

$$REF = \max|\mathbf{\Psi}(\mathbf{s_s})| + A \cdot \sigma_\Psi, \tag{4.1}$$

where $REF$ is the needed reference value; $\mathbf{\Psi}$ is the silence vector, evaluated when TEO is applied to the vector $\mathbf{s_s}$ constituted by audio sampled values for "silence"; $A$ is a constant that depends on SNR[2]; and $\sigma_\Psi$ is the *standard deviation* for TEO values from the silence vector.

The decision if a given frame contains speech or non-speech information is taken by comparison with the reference value. If the maximum absolute TEO value from a given frame *is greater than* the the reference value ($REF$), it

[2]This constant has also dependency on the variability and the complexity of utterances, in a minor degree. SNR is the major factor of dependence.

should contain speech information. Otherwise, it will be considered that this frame contains non-speech information. This inference enables the TSWS method to update reference value every time it gets a non-speech frame. This update is done by excluding samples from the first frame in the beginning of the silence vector, and appending to it the last non-speech frame captured. With this updated silence vector, the TSWS method applies equation (4.1) to reach an updated reference value.

To set the speech boundary inside the captured audio signal, a boolean control variable is set. The TSWS method identifies this variable as *Word* and uses it to keep control of last captured frame status. If this last frame was identified as "speech"[3] and *Word* is *false*, *Word* is set to *true*; if the last frame is considered a non-speech one and *Word* is *true*, *Word* is set to *false*. In other words, if the maximum absolute TEO value from a given frame is *greater than* the reference value and *Word* is *false*, the starting point of this frame is set as the "starting point of a possible spoken word". Reciprocally, if the maximum absolute TEO value from a given frame is *less than* the reference value and *Word* is *true*, the ending point of this frame is set as the "ending point of a possible spoken word".

The TSWS method has also the following adjustments incorporated:

- If a just found "spoken word" boundary is too short to be a phoneme, that boundary is discarded. The minimum lasting time inside boundary is identified as $T_{min}$.

- If the silence after a recently bounded "spoken word" is too short to mean the whole "spoken word" is bounded, the ending point of that boundary is discarded and *Word* is set to *true*. This means the method

[3]Note that not always a relative high absolute value for TEO in a given frame means it carries speech information. It could also be the capture of an interference.

will carry on until finding a new ending point. The minimum lasting time for silence after a word is identified as $T_{saw}$.

An example of TSWS resultant can be found in Figure 4.2, using the SNR-dependent constant $A$ equals 9. This SNR-dependent constant is adjusted empirically, according to the available dataset.

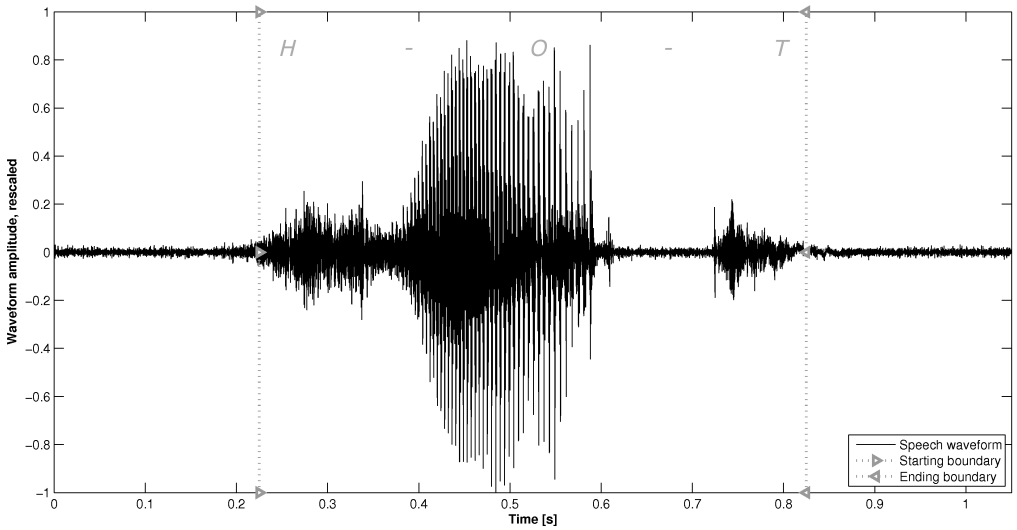



Figure 4.2: Audio waveform for the English word "HOT" /hat/ from support database with white noise addition (SNR 30dB), and respective boundary found by the TSWS method, with $A = 9$.

Table 4.1 presents time parameters for the TSWS method, adjusted to the support database. The full algorithm for TSWS is presented as a flowchart in Figure 4.3 and as pseudocode in the sequence of Algorithms 4.1, 4.2 and 4.3.

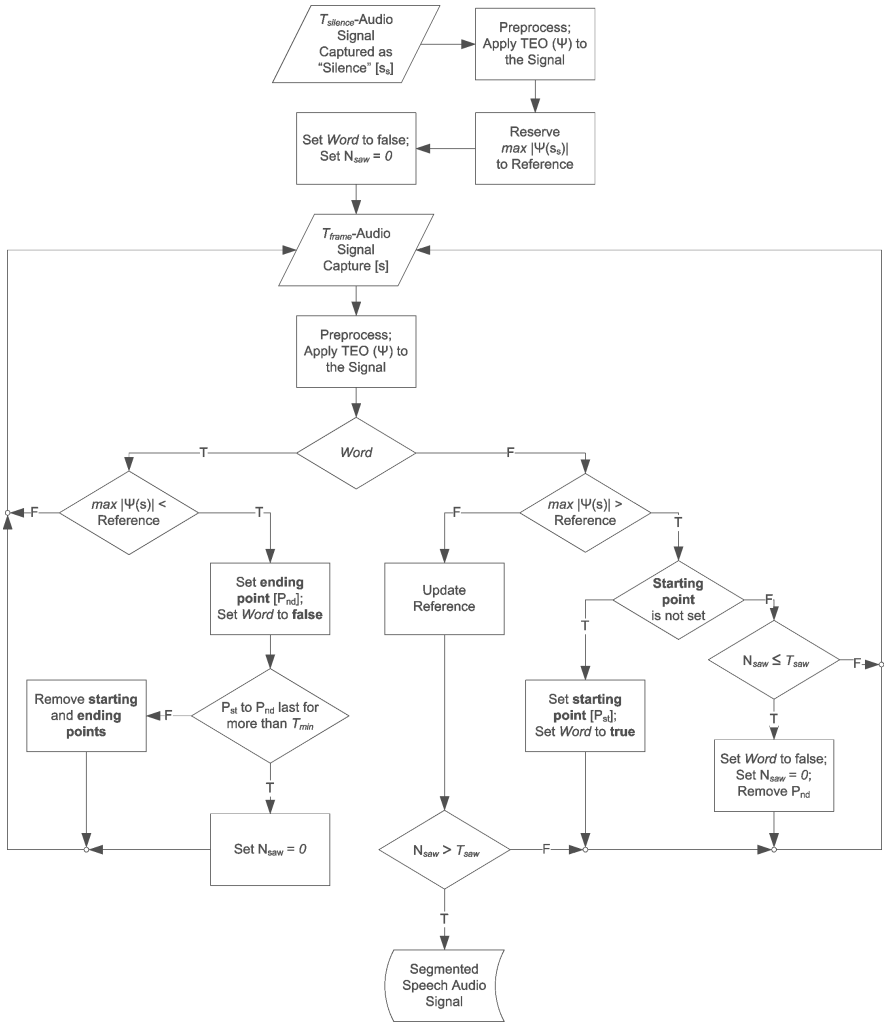



Figure 4.3: Proposed word boundary detection algorithm (flowchart).

Table 4.1: Time parameters (constants) for the TSWS algorithm.

| **Constant** | **Symbol** | **Value** | **# frames** | **# samples** |
|---|---|---|---|---|
| Size of initial silence vector | $T_{silence}$ | 100ms | 4 | 800 |
| Size of non-overlap frame | $T_{frame}$ | 25ms | 1 | 200 |
| Minimum length for valid boundary | $T_{min}$ | 150ms | 6 | 1200 |
| Min. length for silence after boundary's ending point | $T_{saw}$ | 250ms | 10 | 2000 |

**Algorithm 4.1** Proposed TSWS algorithm (pseudocode) - part 1/3

**Require:** $A, Fs$(sample frequency)$, T_{silence}, T_{frame}, T_{min}, T_{saw}$

1: // *** "Silence" capture (transitory state) ***
2: $nsil \leftarrow \frac{F_s}{1000} * T_{silence}$ // $nsil$ is the number of samples for silence, $T_{silence}$ is declared in milliseconds
3: $sil \leftarrow$ capture$(audio, nsil)$ // Capture $nsil$ samples of audio signal
4: // *** Preprocess / Apply TEO ***
5: $spe \leftarrow sof \leftarrow sil$
6: **for** $i = 2$ to $nsil$ **do**
7: $\quad sof[i] \leftarrow sil[i] - sil[i-1] + 0.999 * sof[i-1]$ // Offset compensation
8: $\quad spe[i] \leftarrow sof[i] - 0.97 * sof[i-1]$ // Pre-emphasis
9: **end for**
10: $vosil[1] \leftarrow 0.0$ // Array to save signal's TEO values when in "silence"
11: $vosil[nsil] \leftarrow 0.0$
12: **for** $i = 2$ to $nsil - 1$ **do**
13: $\quad vosil[i] \leftarrow spe[i]^2 - spe[i-1] * spe[i+1]$ // Applying TEO
14: **end for**
15: // *** Reserve reference REF ***
16: $REF =$ maxabs$(vosil)+A*$stdev$(vosil)$ // Maximum absolute value plus $A$ times standard deviation

**Algorithm 4.2** Proposed TSWS algorithm (pseudocode) - part 2/3

17: // *** Preparing permanent processing state ***
18: $word \leftarrow$ false // Control for spoken word state
19: $N_{saw} \leftarrow 0$ // Control for number of samples of "silence" after detected spoken word
20: $nfrm \leftarrow \frac{F_s}{1000} * T_{frame}$ // $nfrm$ is the number of samples for a frame, $T_{frame}$ [ms]
21: $stopcapture \leftarrow$ false // Control variable
22: // *** Audio signal capture (permanent processing state) ***
23: **while** NOT($stopcapture$) **do**
24: $sfr \leftarrow$ capture($audio, nfrm$) // Capture $nfrm$ samples of audio signal
25: // *** Preprocess / Apply TEO ***
26: $spe \leftarrow sof \leftarrow sfr$
27: **for** $i = 2$ to $nfrm$ **do**
28: $sof[i] \leftarrow sfr[i] - sfr[i-1] + 0.999 * sof[i-1]$ // Offset compensation
29: $spe[i] \leftarrow sof[i] - 0.97 * sof[i-1]$ // Pre-emphasis
30: **end for**
31: clear($TEO$) // Clear TEO array
32: $TEO[1] \leftarrow 0.0$ // Array to save signal's TEO values in this frame
33: $TEO[nfrm] \leftarrow 0.0$
34: **for** $i = 2$ to $nfrm - 1$ **do**
35: $TEO[i] \leftarrow spe[i]^2 - spe[i-1] * spe[i+1]$ // Applying TEO
36: **end for**
37: // *** Reserve value for comparison ***
38: $maxTEO \leftarrow$ maxabs($TEO$) // Maximum absolute value

**Algorithm 4.3** Proposed TSWS algorithm (pseudocode) - part 3/3

```
39:     // *** Decision for frame: speech or "silence" ***
40:     if word is true then
41:         if maxTEO < REF then
42:             P_nd ← frame id // Set ending point
43:             word ← false
44:             // If detected spoken word lasts for more than T_min [ms]
45:             if P_nd − P_st > (F_s/1000) * T_min then
46:                 N_saw ← 0
47:             else
48:                 P_st ← P_nd ← ∅ // Delete starting and ending points
49:             end if
50:         end if
51:     else[word is false]
52:         N_saw ← N_saw + nfrm
53:         if maxTEO > REF then
54:             if isempty(P_st) then
55:                 // If silence after word is less than T_saw [ms]
56:                 if N_saw ≤ (F_s/1000) * T_saw then
57:                     word ← false
58:                     N_saw ← 0
59:                     P_nd ← ∅ // Delete ending points
60:                 end if
61:             else
62:                 P_st ← frame id // Set starting point
63:                 word ← true
64:             end if
65:         else
66:             // *** Update REF ***
67:             vosil ← vosil[nfrm + 1 : end] ⊕ TEO // Concatenation
68:             REF = maxabs(vosil) + A∗stdev(vosil)
69:             // If silence after word lasts for more than T_saw [ms]
70:             if N_saw > (F_s/1000) * T_saw then
71:                 // *** Segmented Speech Audio Signal ***
72:                 stopcapture ← true
73:             end if
74:         end if
75:     end if
76: end while
```

## 4.3 TSWS Experimental Results

In order to evaluate the performance of the TSWS method, two other speech segmentation methods were implemented, for purposes of comparison: Classical and Bottom-up methods. Classical method uses energy and zero-crossing rate computations [56], in order to detect the beginning and the ending of a spoken word present in a given audio signal. Bottom-up method, also known as Hybrid Endpoint Detector, was proposed by Lamel, Rabiner, et al [35] and uses concepts of adaptive level equalization, energy pulse detection, and ordering of found boundary.

All mentioned methods were applied to the 258 audio files from the support database and their respective detected boundaries were saved. To enable comparison, all 258 audio signals passed through a manual segmentation, i.e., each audio signal was plotted and their respective spoken words boundaries were detected by human interaction. Even that this process is not exact, we disregarded involved errors and assumed those manually achieved boundaries to be the targets for the methods here considered. We have considered the evaluation for the starting points separately from the evaluation of the ending points, in a first moment. When we joined both starting and ending points results, we call these as overall results.

Assuming manual detected boundaries as targets, the boundaries found by the three methods were compared to those targets by taking the difference in seconds between them. Figure 4.4 illustrates this kind of comparison. Note that both Classical and Bottom-up methods miss the nasal /n/. Classical has also got confused with the starting point setting. The TSWS method, in the other hand, has set the boundary precisely.

Then, the differences between target and found boundaries for all methods were used to evaluate the root mean square error (RMSE) of each

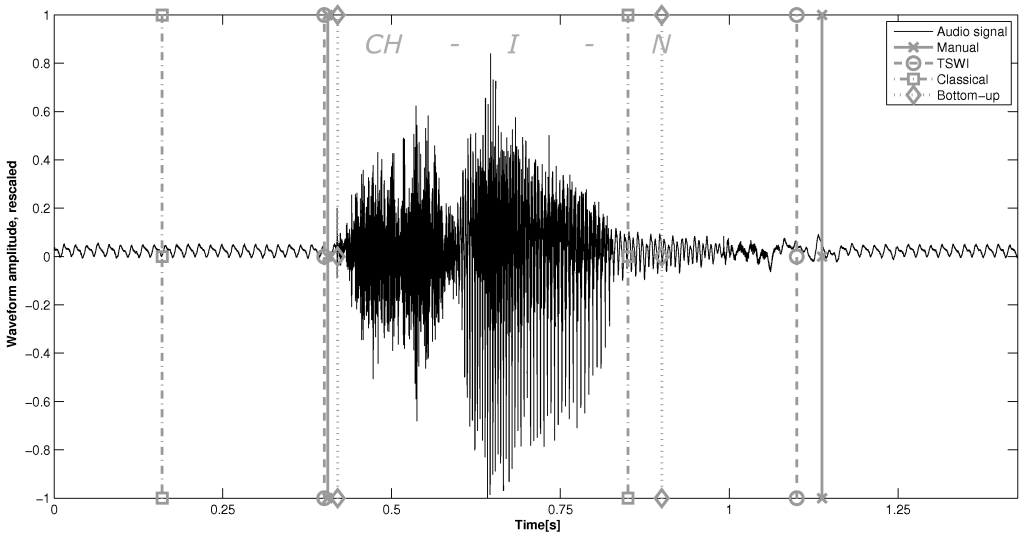



Figure 4.4: Audio waveform for the English word "CHIN" /tʃɪn/, target manually positioned boundary, and respective boundaries found by TSWS, Classical and Bottom-up methods.

method. The RMSE, also known as root mean square deviation, is an often-used measure of the differences between values predicted by a model or an estimator and the values actually observed in reality (target values). It is defined by the square root of the mean square error, as presented in equation (4.2). The choice of using RMSE was taken because it keeps the original unit from the involved quantities. In the present case, RMSE will keep track of the error in seconds.

$$\begin{aligned} mse &= \frac{1}{N} \cdot \sum_{i=1}^{N} e(i)^2 = \frac{1}{N} \cdot \sum_{i=1}^{N} (t(i) - a(i))^2 \\ rmse &= \sqrt{mse}, \end{aligned} \tag{4.2}$$

where $mse$ is the mean square error; $N$ is the number of patterns; $e(i)$ is the respective error of the i$^{th}$ pattern; $t(i)$ is the target (or expected) value for

the i$^{th}$ pattern; $a(i)$ is the actual value (or estimator) for the i$^{th}$ pattern; and $rmse$ is the root mean square error.

One of the reasons for using support database for the TSWS method evaluation is that support database includes several (or almost all) phoneme sounds from American English. Trying to extend this evaluation, we have divided RMSE through 14 basic types of phonemes, presented by support database. So, TSWS, Classical and Bottom-up methods were tried with *3-element blend*, *affricate*, *back vowel*, *central vowel*, *diphthong*, *friction*, *front vowel*, *glide*, *l-blend*, *liquid*, *nasal*, *r-blend*, *s-blend*, and *stop* sounds. Overall RMSE values means that results from all those 14 basic types of phonemes were considered in the calculation.

Another aspect considered during the preparation of this evaluation was to explore how those three segmentation methods react when facing different levels of noise. Actually, due to Lombard reflex[4] [28], no noise artificially added to a speech signal reflects the reality. This research tries to minimize this fact by artificially adding white Gaussian noise (WGN) to the audio signals from support database. The noise added this way could reflect a possible noise added by the transducer used to capture that audio signal. All above mentioned speech segmentation methods were also tested in the same SNR conditions. The conducted tests used the 258 audio files with WGN addition reaching a SNR of 5dB, 15dB, and 30dB, besides the original support database that presents a SNR great above 30dB.

The TSWS method, different from the others, has an SNR-dependent constant to be empirically adjusted. Table 4.2 shows the best empirically adjusted constant values for each SNR level.

[4]The Lombard reflex (or Lombard effect) is a noise-induced stress phenomenon that yields a modification of the speaker speech production in the presence of adverse conditions such as noise.

Table 4.2: Empirical SNR-dependent constant $A$ from the TSWS method against SNR.

| | **SNR (WGN addition)** | | | |
|---|---|---|---|---|
| | **Clear** | **30dB** | **15dB** | **5dB** |
| constant $A$ | 25 | 9 | 3 | 1.1 |

Figure 4.5 presents the curve obtained by interpolation [18] of empirical found values for SNR-dependent constant $A$. It was assumed a sigmoidal appearance for $A$, because of minimum and maximum values experimented (1.1 and 25, respectively). This curve intends to be a good estimator for empirical SNR-dependent constant $A$, based on conducted tests.

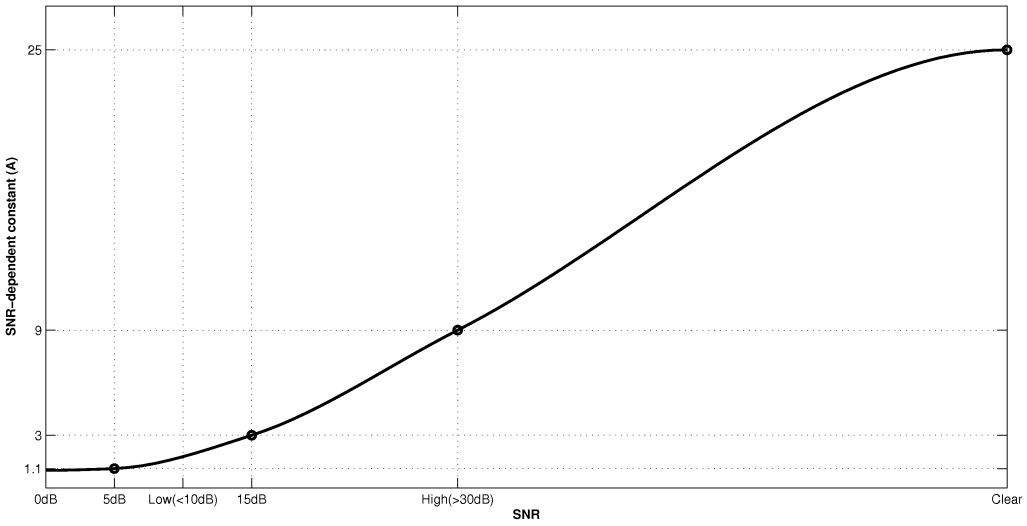



Figure 4.5: Estimator curve for empirical SNR-dependent constant $A$.

Results from the conducted tests can be found in Table 4.3. More discriminated results can be found in Tables 4.4, 4.5, 4.6, and 4.7 from Section 4.4. Figures 4.6, 4.7, 4.8 and 4.9 translate numerical results into graphical ones.

As one can see, the TSWS method presents the best of all overall RMSE values, indicating a better performance of all compared segmentation methods. TSWS method decreases the RMSE on the spoken word boundary detection to 32.2% of Classical method performance, and to 38.8% of

Table 4.3: Overall RMSE (in milliseconds) from TSWS, Classical, and Bottom-up segmentation methods.

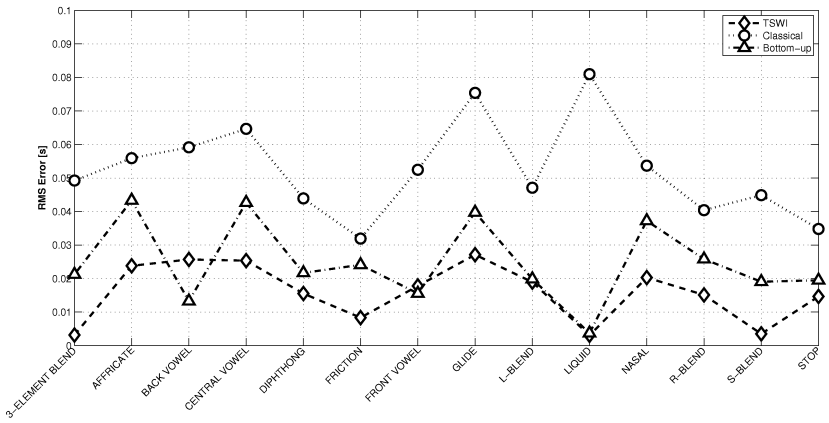
| | **SNR (WGN addition)** | | | |
|---|---|---|---|---|
| | **Clear** | **30dB** | **15dB** | **5dB** |
| | **RMSE [ms]** | | | |
| TSWS | 3.8 | 3.7 | 7.4 | 10.5 |
| Classical | 11.8 | 12.6 | 11.9 | 14.4 |
| Bottom-up | 9.8 | 11.1 | 14.6 | 20.4 |

Bottom-up method performance on noise-clear audio signals[5]. This means TSWS performance increased the precision on boundary detection, reducing the RMSE of 67.8% when comparing to Classical method, and of 61.2%, when comparing to Bottom-up method. Note that the worst performance from conducted tests for the TSWS method, when detecting boundary from audio signals with a SNR of 5dB, is still better than the best Classical segmentation method performance, when applied to the original audio signal. In other words, in the worst performance case from conducted tests, the TSWS method decreases the RMSE on boundary detection to 72.9% of Classical method performance, and to 51.5% of Bottom-up method performance on the cases where we have a SNR of 5dB. This means an error reduction of 27.1% and 48.5%, respectively. Curiously, Bottom-up loses in performance to Classical method when SNR $\leq$ 15dB, fact that was not expected due to the better performance of Bottom-up in higher SNRs. Those overall RMSE results mean that, in the cases of wrong boundary detection, the TSWS method misses the target by little when compared to the others.

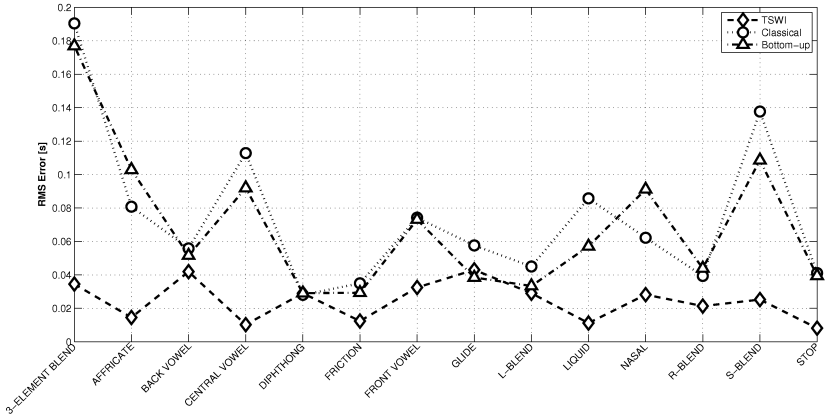
Regarding intrinsic difficulties per type of phonemes, *friction* and *stop* sounds were the ones which all methods had their best individual perfor-

[5]The very small improvement in performance of the TSWS method perceived in 30dB when compared with clear signal performance suggests that the TSWS time parameters are also sensitive to noise. Although, this sensitivity presents itself at a very much lower level than the sensitivity to noise of the constant $A$. This is the main reason we have kept those parameters as constants, even when dealing with different SNR's.

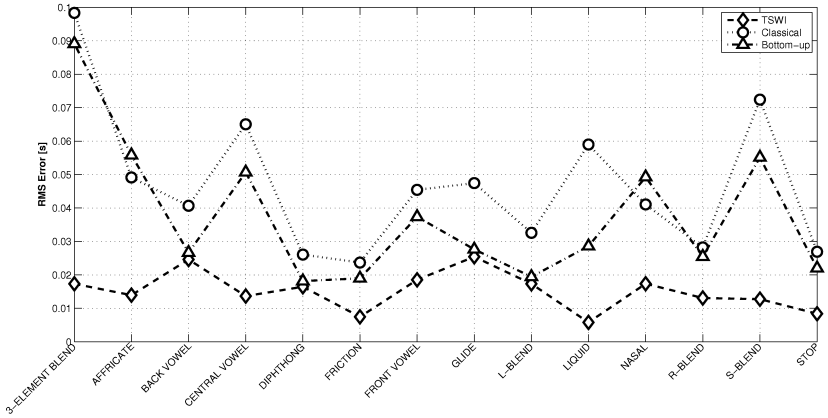
mances. TSWS got an excellent performance with *liquids*, while Classical and Bottom-up had very good individual performances with *l-blends* and *r-blends*, respectively, and both with *diphthongs*. Worst cases for TSWS were *back vowels* and *glides*, while for Classical and Bottom-up had worst performances for *3-element blends*, mainly for reaching an ending point too early and missing the rest of the word. In the lowest SNR experiment, *frictions* and *stops* kept all methods best individual performances. *Liquids* turned themselves as the worst individual performance case for all methods.




(a) Starting point (first boundary) error.

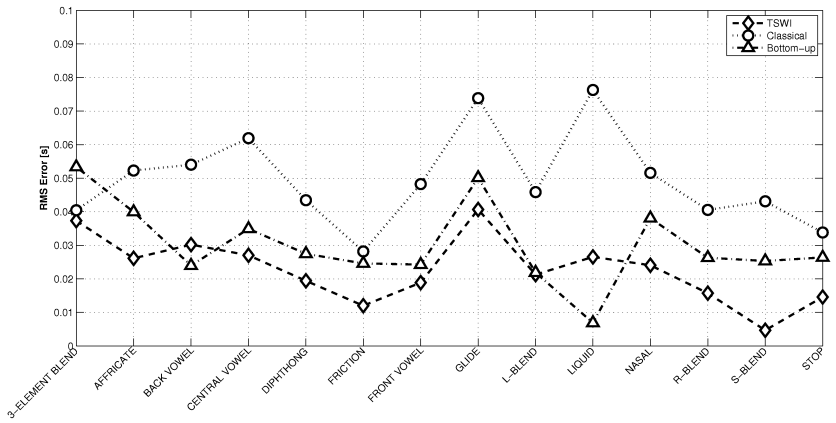



(b) Ending point (last boundary) error.

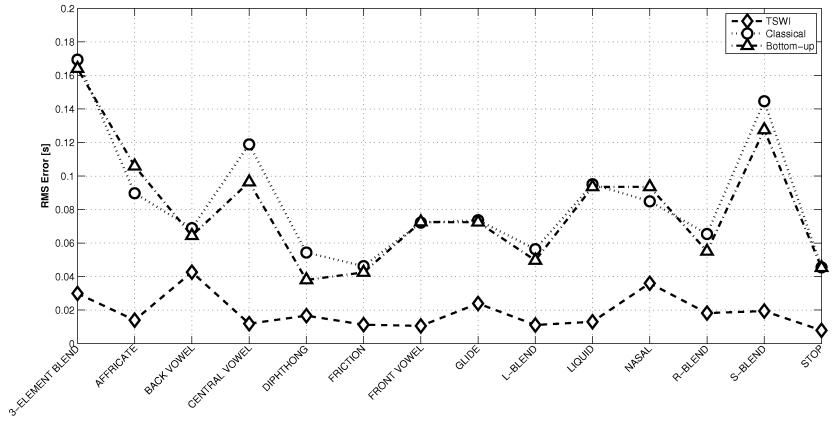



(c) Overall error.

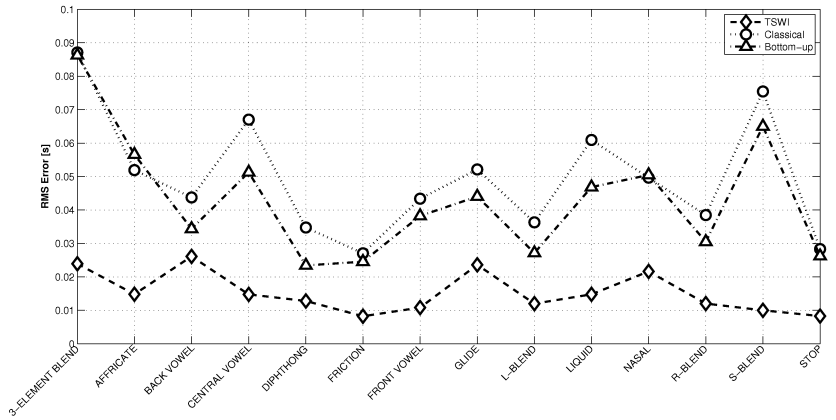
Figure 4.6: RMSE per phoneme type for TSWS ($A = 25$), Classical and Bottom-up methods, with clear signals.




(a) Starting point (first boundary) error.

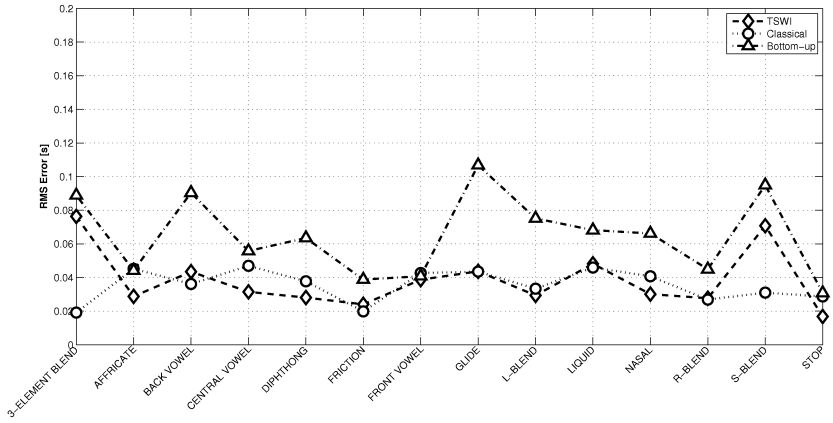



(b) Ending point (last boundary) error.

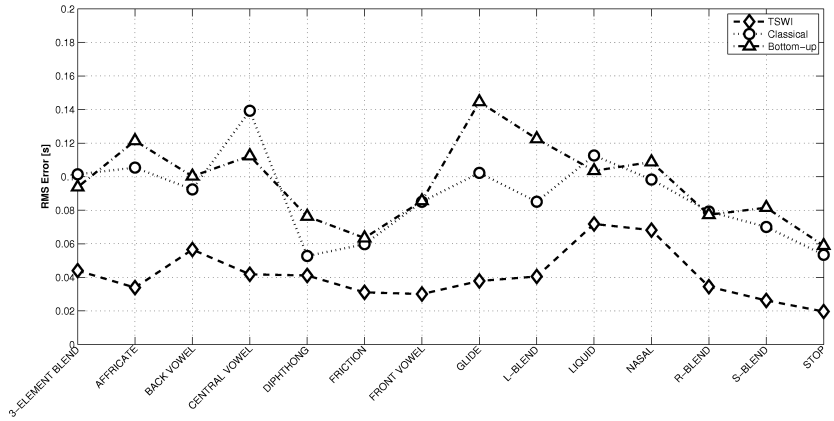



(c) Overall error.

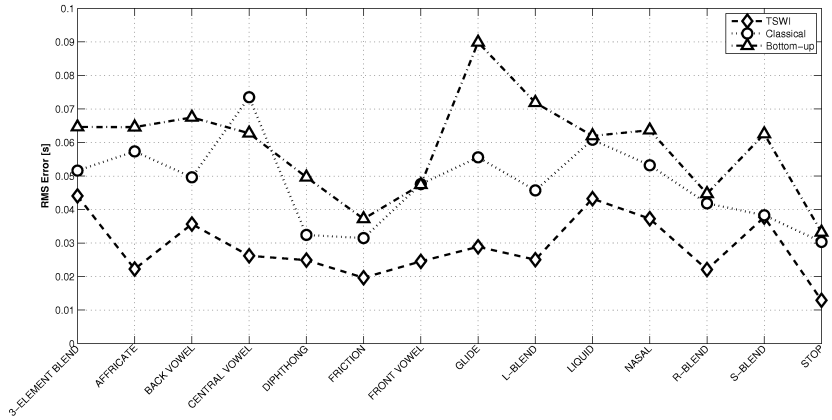
Figure 4.7: RMSE per phoneme type for TSWS ($A = 9$), Classical and Bottom-up methods, with $SNR = 30dB$.




(a) Starting point (first boundary) error.

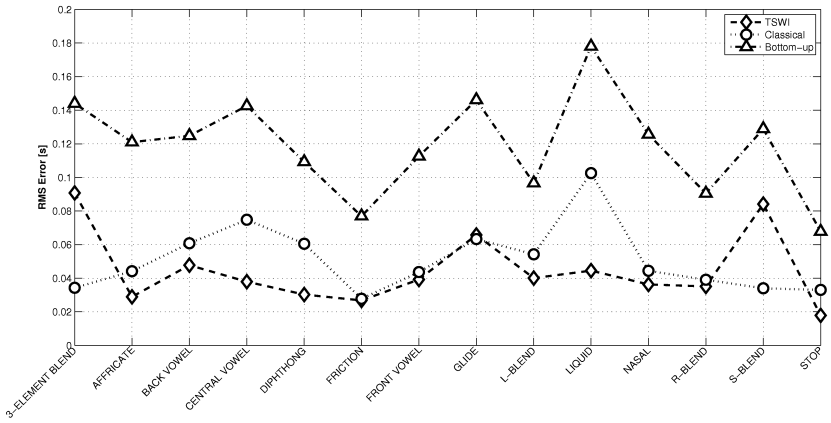



(b) Ending point (last boundary) error.

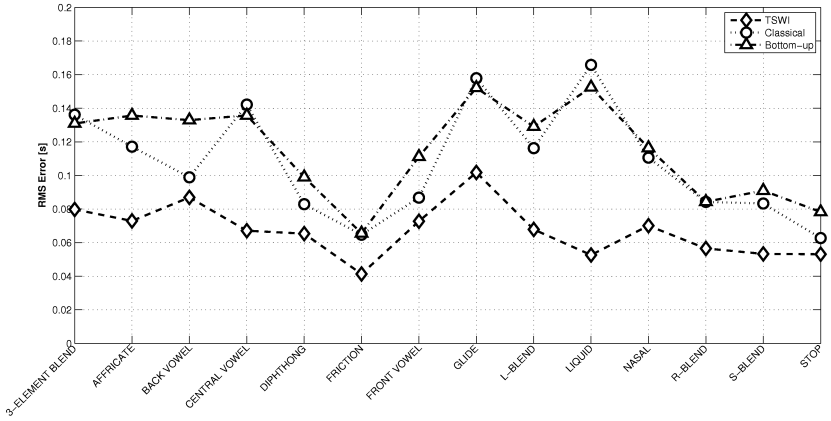



(c) Overall error.

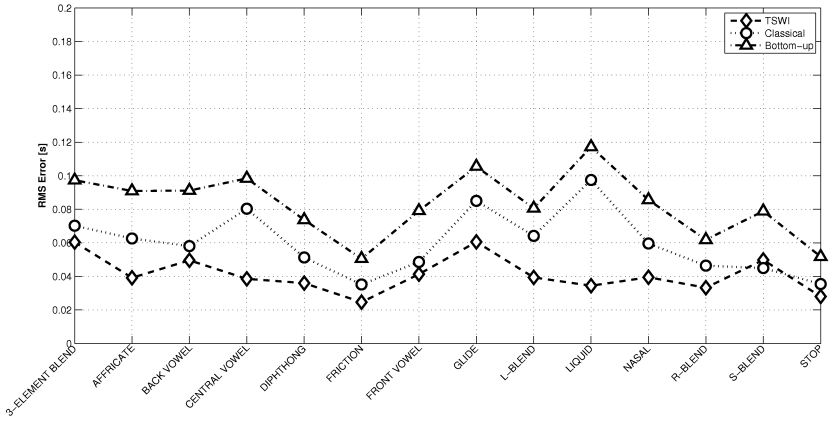
Figure 4.8: RMSE per phoneme type for TSWS ($A = 3$), Classical and Bottom-up methods, with $SNR = 15dB$.




(a) Starting point (first boundary) error.




(b) Ending point (last boundary) error.




(c) Overall error.

Figure 4.9: RMSE per phoneme type for TSWS ($A = 1.1$), Classical and Bottom-up methods, with $SNR = 5dB$.

## 4.4 Extended Comparison between Methods

This section presents the tables containing all data regarding root mean square error (RMSE) from each speech segmentation method compared — TSWS, Classical, and Bottom-up — when detecting the boundary (starting and ending points) of a given spoken word. The "overall" column register the overall RMSE considering both starting and ending detection. Results presented here are divided by the types of American English phonemes, as in Compton's work [10]. Conducted tests were performed with the artificial addition of white Gaussian noise to each audio file from this support database, reaching different SNR for comparison purposes only. Each of the following tables presents results of the same dataset with a different SNR.

Table 4.4: Support database (Clear signal) comparison of TSWS ($A = 25$) with Classical and Bottom-up methods.

| Phoneme Type | Method | Root Mean Square Error [s] | | |
|---|---|---|---|---|
| | | Starting Point | Ending Point | Overall |
| 3-ELEMENT BLEND | TSWI | 0.003163 | 0.034504 | 0.017324 |
| | Classic | 0.049241 | 0.19043 | 0.098346 |
| | Bottom-up | 0.021244 | 0.17696 | 0.089113 |
| AFFRICATE | TSWI | 0.023809 | 0.014535 | 0.013948 |
| | Classic | 0.055929 | 0.080792 | 0.049131 |
| | Bottom-up | 0.043338 | 0.1029 | 0.055825 |
| BACK VOWEL | TSWI | 0.025695 | 0.041939 | 0.024592 |
| | Classic | 0.059152 | 0.055841 | 0.040673 |
| | Bottom-up | 0.013258 | 0.051599 | 0.026638 |
| CENTRAL VOWEL | TSWI | 0.025343 | 0.010256 | 0.01367 |
| | Classic | 0.064667 | 0.11284 | 0.065027 |
| | Bottom-up | 0.042658 | 0.091983 | 0.050697 |
| DIPHTHONG | TSWI | 0.015532 | 0.02901 | 0.016453 |
| | Classic | 0.04392 | 0.028091 | 0.026067 |
| | Bottom-up | 0.021682 | 0.029086 | 0.018139 |
| FRICTION | TSWI | 0.0083389 | 0.012389 | 0.0074672 |
| | Classic | 0.031926 | 0.034956 | 0.023671 |
| | Bottom-up | 0.024107 | 0.029314 | 0.018976 |
| FRONT VOWEL | TSWI | 0.017813 | 0.032478 | 0.018521 |
| | Classic | 0.052466 | 0.0742 | 0.045438 |
| | Bottom-up | 0.015547 | 0.073057 | 0.037346 |
| GLIDE | TSWI | 0.027122 | 0.043034 | 0.025434 |
| | Classic | 0.075404 | 0.057641 | 0.047456 |
| | Bottom-up | 0.039752 | 0.038436 | 0.027647 |
| L-BLEND | TSWI | 0.018948 | 0.029005 | 0.017323 |
| | Classic | 0.047089 | 0.045011 | 0.03257 |
| | Bottom-up | 0.019879 | 0.033342 | 0.019409 |
| LIQUID | TSWI | 0.0030751 | 0.011165 | 0.0057904 |
| | Classic | 0.080983 | 0.085772 | 0.058981 |
| | Bottom-up | 0.0037078 | 0.057054 | 0.028587 |
| NASAL | TSWI | 0.020286 | 0.028184 | 0.017363 |
| | Classic | 0.05368 | 0.062197 | 0.041079 |
| | Bottom-up | 0.037233 | 0.091171 | 0.04924 |
| R-BLEND | TSWI | 0.015101 | 0.021362 | 0.013081 |
| | Classic | 0.040394 | 0.039391 | 0.028211 |
| | Bottom-up | 0.025807 | 0.0439 | 0.025462 |
| S-BLEND | TSWI | 0.0035127 | 0.025241 | 0.012742 |
| | Classic | 0.044871 | 0.13767 | 0.072401 |
| | Bottom-up | 0.019046 | 0.1086 | 0.055127 |
| STOP | TSWI | 0.014667 | 0.008154 | 0.0083907 |
| | Classic | 0.03479 | 0.041094 | 0.026921 |
| | Bottom-up | 0.019497 | 0.039492 | 0.022021 |
| OVERALL | TSWI | | | 0.003771 |
| | Classic | | | 0.011806 |
| | Bottom-up | | | 0.0097696 |

Table 4.5: Modified support database ($SNR = 30dB$) comparison of TSWS ($A = 9$) with Classical and Bottom-up methods.

| Phoneme Type | Method | Root Mean Square Error [s] | | |
|---|---|---|---|---|
| | | Starting Point | Ending Point | Overall |
| 3-ELEMENT BLEND | TSWI | 0.037331 | 0.029909 | 0.023917 |
| | Classic | 0.040439 | 0.16945 | 0.087105 |
| | Bottom-up | 0.053349 | 0.16409 | 0.086273 |
| AFFRICATE | TSWI | 0.026118 | 0.013966 | 0.014809 |
| | Classic | 0.052289 | 0.089712 | 0.051919 |
| | Bottom-up | 0.039959 | 0.1059 | 0.056592 |
| BACK VOWEL | TSWI | 0.030215 | 0.042618 | 0.026121 |
| | Classic | 0.054032 | 0.068865 | 0.043766 |
| | Bottom-up | 0.023955 | 0.06441 | 0.03436 |
| CENTRAL VOWEL | TSWI | 0.02701 | 0.011882 | 0.014754 |
| | Classic | 0.061929 | 0.11888 | 0.067021 |
| | Bottom-up | 0.034911 | 0.096385 | 0.051256 |
| DIPHTHONG | TSWI | 0.019421 | 0.016662 | 0.012795 |
| | Classic | 0.043428 | 0.054317 | 0.034772 |
| | Bottom-up | 0.027475 | 0.037988 | 0.023441 |
| FRICTION | TSWI | 0.01199 | 0.01128 | 0.008231 |
| | Classic | 0.028165 | 0.046197 | 0.027053 |
| | Bottom-up | 0.024611 | 0.0425 | 0.024556 |
| FRONT VOWEL | TSWI | 0.018879 | 0.010548 | 0.010813 |
| | Classic | 0.048232 | 0.072155 | 0.043395 |
| | Bottom-up | 0.024241 | 0.072518 | 0.038231 |
| GLIDE | TSWI | 0.040671 | 0.023925 | 0.023593 |
| | Classic | 0.073844 | 0.0736 | 0.052129 |
| | Bottom-up | 0.05012 | 0.072397 | 0.044027 |
| L-BLEND | TSWI | 0.021258 | 0.011089 | 0.011988 |
| | Classic | 0.045853 | 0.056305 | 0.036307 |
| | Bottom-up | 0.021891 | 0.049656 | 0.027134 |
| LIQUID | TSWI | 0.026539 | 0.013019 | 0.01478 |
| | Classic | 0.076287 | 0.095108 | 0.060961 |
| | Bottom-up | 0.0068942 | 0.093435 | 0.046845 |
| NASAL | TSWI | 0.024059 | 0.035998 | 0.021649 |
| | Classic | 0.051573 | 0.084878 | 0.049659 |
| | Bottom-up | 0.038077 | 0.093538 | 0.050496 |
| R-BLEND | TSWI | 0.015724 | 0.018198 | 0.012025 |
| | Classic | 0.040545 | 0.065389 | 0.038469 |
| | Bottom-up | 0.026294 | 0.055037 | 0.030498 |
| S-BLEND | TSWI | 0.0046761 | 0.019426 | 0.0099906 |
| | Classic | 0.043105 | 0.14462 | 0.075456 |
| | Bottom-up | 0.025321 | 0.12752 | 0.065005 |
| STOP | TSWI | 0.014575 | 0.007881 | 0.0082846 |
| | Classic | 0.033815 | 0.045445 | 0.028323 |
| | Bottom-up | 0.026387 | 0.045336 | 0.026228 |
| OVERALL | TSWI | | | 0.003669 |
| | Classic | | | 0.01263 |
| | Bottom-up | | | 0.011125 |

Table 4.6: Modified support database ($SNR = 15dB$) comparison of TSWS ($A = 3$) with Classical and Bottom-up methods.

| Phoneme Type | Method | Root Mean Square Error [s] | | |
|---|---|---|---|---|
| | | Starting Point | Ending Point | Overall |
| 3-ELEMENT BLEND | TSWI | 0.07634 | 0.044004 | 0.044057 |
| | Classic | 0.01917 | 0.10143 | 0.051613 |
| | Bottom-up | 0.08892 | 0.093772 | 0.064614 |
| AFFRICATE | TSWI | 0.028778 | 0.033926 | 0.022244 |
| | Classic | 0.045159 | 0.10541 | 0.057339 |
| | Bottom-up | 0.044167 | 0.12141 | 0.064595 |
| BACK VOWEL | TSWI | 0.043495 | 0.056553 | 0.035673 |
| | Classic | 0.036168 | 0.092386 | 0.049606 |
| | Bottom-up | 0.090424 | 0.1002 | 0.067485 |
| CENTRAL VOWEL | TSWI | 0.031484 | 0.041846 | 0.026184 |
| | Classic | 0.046956 | 0.1393 | 0.073498 |
| | Bottom-up | 0.055791 | 0.11239 | 0.062737 |
| DIPHTHONG | TSWI | 0.028059 | 0.041145 | 0.024901 |
| | Classic | 0.037745 | 0.052725 | 0.032421 |
| | Bottom-up | 0.063514 | 0.076299 | 0.049638 |
| FRICTION | TSWI | 0.024055 | 0.031063 | 0.019644 |
| | Classic | 0.019866 | 0.05975 | 0.031483 |
| | Bottom-up | 0.038795 | 0.063473 | 0.037195 |
| FRONT VOWEL | TSWI | 0.038753 | 0.030049 | 0.024519 |
| | Classic | 0.042683 | 0.08503 | 0.047571 |
| | Bottom-up | 0.040781 | 0.085571 | 0.047396 |
| GLIDE | TSWI | 0.043692 | 0.037833 | 0.028898 |
| | Classic | 0.043511 | 0.10227 | 0.05557 |
| | Bottom-up | 0.10682 | 0.14456 | 0.089874 |
| L-BLEND | TSWI | 0.029363 | 0.040541 | 0.025029 |
| | Classic | 0.033383 | 0.085092 | 0.045703 |
| | Bottom-up | 0.075168 | 0.12256 | 0.071888 |
| LIQUID | TSWI | 0.048177 | 0.071801 | 0.043233 |
| | Classic | 0.045917 | 0.11262 | 0.060812 |
| | Bottom-up | 0.068199 | 0.1035 | 0.061976 |
| NASAL | TSWI | 0.030105 | 0.068152 | 0.037252 |
| | Classic | 0.040722 | 0.098278 | 0.05319 |
| | Bottom-up | 0.066204 | 0.10875 | 0.06366 |
| R-BLEND | TSWI | 0.027681 | 0.034449 | 0.022096 |
| | Classic | 0.026898 | 0.079257 | 0.041849 |
| | Bottom-up | 0.044915 | 0.077306 | 0.044704 |
| S-BLEND | TSWI | 0.070779 | 0.026222 | 0.03774 |
| | Classic | 0.031016 | 0.070013 | 0.038288 |
| | Bottom-up | 0.094891 | 0.081535 | 0.062554 |
| STOP | TSWI | 0.016798 | 0.01965 | 0.012926 |
| | Classic | 0.028721 | 0.05343 | 0.03033 |
| | Bottom-up | 0.030805 | 0.058898 | 0.033234 |
| OVERALL | TSWI | | | 0.0073755 |
| | Classic | | | 0.011929 |
| | Bottom-up | | | 0.014588 |

Table 4.7: Modified support database ($SNR = 5dB$) comparison of TSWS ($A = 1.1$) with Classical and Bottom-up methods.

| Phoneme Type | Method | Root Mean Square Error [s] | | |
|---|---|---|---|---|
| | | Starting Point | Ending Point | Overall |
| 3-ELEMENT BLEND | TSWI | 0.090763 | 0.079766 | 0.060416 |
| | Classic | 0.034311 | 0.13615 | 0.070205 |
| | Bottom-up | 0.14408 | 0.13094 | 0.097346 |
| AFFRICATE | TSWI | 0.028908 | 0.072895 | 0.039209 |
| | Classic | 0.044179 | 0.11714 | 0.062596 |
| | Bottom-up | 0.12102 | 0.13566 | 0.090899 |
| BACK VOWEL | TSWI | 0.047776 | 0.086927 | 0.049596 |
| | Classic | 0.06081 | 0.0989 | 0.05805 |
| | Bottom-up | 0.12476 | 0.13289 | 0.091138 |
| CENTRAL VOWEL | TSWI | 0.037958 | 0.067088 | 0.038541 |
| | Classic | 0.074816 | 0.14215 | 0.08032 |
| | Bottom-up | 0.1426 | 0.13576 | 0.098443 |
| DIPHTHONG | TSWI | 0.030265 | 0.065368 | 0.036017 |
| | Classic | 0.060495 | 0.082893 | 0.05131 |
| | Bottom-up | 0.10927 | 0.098978 | 0.073715 |
| FRICTION | TSWI | 0.026771 | 0.041324 | 0.024619 |
| | Classic | 0.02776 | 0.064691 | 0.035198 |
| | Bottom-up | 0.076946 | 0.065523 | 0.050532 |
| FRONT VOWEL | TSWI | 0.039239 | 0.072698 | 0.041306 |
| | Classic | 0.043693 | 0.086789 | 0.048583 |
| | Bottom-up | 0.1126 | 0.11111 | 0.079094 |
| GLIDE | TSWI | 0.065767 | 0.10176 | 0.060583 |
| | Classic | 0.063302 | 0.15788 | 0.08505 |
| | Bottom-up | 0.14619 | 0.15214 | 0.1055 |
| L-BLEND | TSWI | 0.040103 | 0.06782 | 0.039395 |
| | Classic | 0.054258 | 0.11624 | 0.06414 |
| | Bottom-up | 0.096617 | 0.12905 | 0.080605 |
| LIQUID | TSWI | 0.044497 | 0.052632 | 0.034461 |
| | Classic | 0.10256 | 0.16582 | 0.097485 |
| | Bottom-up | 0.17806 | 0.1525 | 0.11722 |
| NASAL | TSWI | 0.03631 | 0.070083 | 0.039466 |
| | Classic | 0.044431 | 0.1106 | 0.059597 |
| | Bottom-up | 0.12563 | 0.11639 | 0.085631 |
| R-BLEND | TSWI | 0.035085 | 0.056524 | 0.033264 |
| | Classic | 0.039077 | 0.084216 | 0.04642 |
| | Bottom-up | 0.090496 | 0.084312 | 0.061843 |
| S-BLEND | TSWI | 0.084124 | 0.053303 | 0.049795 |
| | Classic | 0.034028 | 0.083287 | 0.044985 |
| | Bottom-up | 0.12892 | 0.090979 | 0.078894 |
| STOP | TSWI | 0.017881 | 0.053099 | 0.028015 |
| | Classic | 0.033055 | 0.062703 | 0.035441 |
| | Bottom-up | 0.067845 | 0.078257 | 0.051786 |
| OVERALL | TSWI | | | 0.010468 |
| | Classic | | | 0.014363 |
| | Bottom-up | | | 0.020389 |

# Chapter 5

# Experimental Results

## 5.1 Training and Testing Patterns

The project database is composed of 50 different spoken word records with three utterances each from 69 different speakers, as can be seen in Section 1.3. This total of 10,350 recorded patterns was arbitrarily separated into two groups:

- Training patterns: 6,900 ($1^{st}$ and $2^{nd}$ utterances of each spoken word from each speaker)
- Testing patterns: 3,450 ($3^{rd}$ utterance of each spoken word from each speaker)

The training patterns were used in the supervised training of the project recognizer. After, they were used to test recognizer's learning capability. The testing patterns were used after training, to test recognizer's robustness when faced to "nonlearned" patterns. Besides Bayesian Regularization as a

stop-training decider, a maximum of 1,000 epochs was set to limit project ANN training process.

## 5.2 Results from Project Database

The resultant successful recognition rates presented here were found as the bests of a ANN training series. The conducted tests have used TSWS, Classical and Bottom-up methods in order to compare the contribution of the proposed segmentation method to the recognizer.

A successful recognition means the output neuron that represents the respective target class (command) of a given input pattern is strongly activated when compared to the others. The maximum output level is here considered to be the active output neuron. The division of training and testing patterns (see Section 5.1) was kept here in order to evaluate the learning capability and the ability of generalization from the project recognizer.

Table 5.1 presents the overall successful recognition rates of the project recognizer (MLP) using 1,280 MFC coefficients as input patterns. Results from Classical and Bottom-up methods have shown no difference in recognition rates.

Table 5.1: Overall successful recognition rates (in %) for MFCC-MLP-recognizer.

| **Classical** | | **Bottom-up** | | **TSWS** | |
|---|---|---|---|---|---|
| **Train** | **Test** | **Train** | **Test** | **Train** | **Test** |
| 99.8 | 98.0 | 99.8 | 98.0 | 99.9 | 98.1 |

Because we are too much close to 100% of successful recognition rates, minor increases on successful rates could be considered as huge contributions. As one can see, the best performance of TSWS on spoken word boundary detection had increased the project recognizer's overall successful rates by

0.1%. This means an error reduction of 5%. Figures 5.1 and 5.2 present successful recognition rates for each class of voice command, using Classical, Bottom-up and TSWS methods to support the MFCC-MLP-recognizer. Note that, as Classical and Bottom-up methods have shown the same results from a 50 words vocabulary, they are represented in those figures as the same curve.

To better understand the data plotted on those figures, Tables 5.2 and 5.3 show the worse cases and the better cases of using TSWS method compared to Classical and Bottom-up methods. When we add totals and divide by the number of existing commands (50 words), we reach the increase of 0.096% (rounded to 0.1%) mentioned before.

Table 5.2: Worse TSWS supported individual recognition rates (RR), in %, when compared to Classical and Bottom-up (CL/BU) methods.

| **ID** | **Command** | **TSWS RR** | **CL/BU RR** | $\Delta_{RR}$ |
|---|---|---|---|---|
| 3 | TRÊS | 98.6 | 100.0 | -1.4 |
| 6 | SEIS | 95.7 | 97.10 | -1.4 |
| 18 | CERTO | 92.8 | 95.7 | -2.9 |
| 25 | NORDESTE | 98.6 | 100.0 | -1.4 |
| 36 | ÁRIES | 98.6 | 100.0 | -1.4 |
| 43 | ESCORPIÃO | 94.2 | 95.7 | -1.5 |
| 49 | AJUDA | 97.1 | 98.6 | -1.5 |
| **Total** | | | | **-11.5** |

Table 5.3: Better TSWS supported individual recognition rates (RR), in %, when compared to Classical and Bottom-up (CL/BU) methods.

| ID | Command | TSWS RR | CL/BU RR | $\Delta_{RR}$ |
|---|---|---|---|---|
| 0 | ZERO | 98.6 | 97.1 | 1.5 |
| 2 | DOIS | 97.1 | 95.7 | 1.4 |
| 9 | NOVE | 94.2 | 91.3 | 2.9 |
| 12 | NÃO | 95.7 | 94.2 | 1.5 |
| 17 | AVANÇAR | 98.6 | 97.1 | 1.5 |
| 21 | DÓLAR | 98.6 | 97.1 | 1.5 |
| 30 | DEPARTAMENTO | 98.6 | 97.1 | 1.5 |
| 37 | TOURO | 98.6 | 97.1 | 1.5 |
| 45 | SAGITÁRIO | 98.6 | 97.1 | 1.5 |
| 47 | PEIXES | 95.7 | 94.2 | 1.5 |
| **Total** | | | | **16.3** |

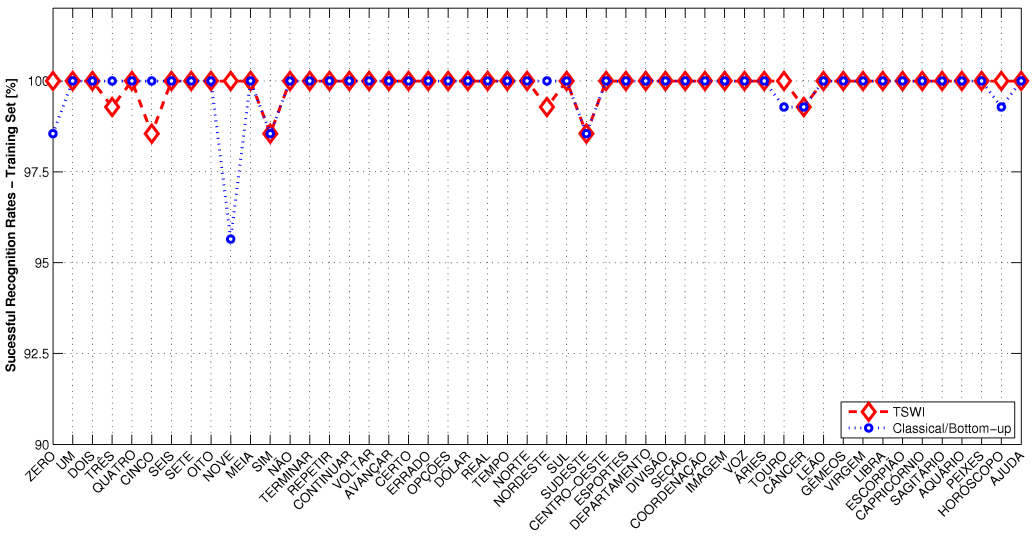



Figure 5.1: Comparison of the training set successful recognition rates, in %, for MFCC-MLP-recognizer using TSWS, Classical and Bottom-up methods.

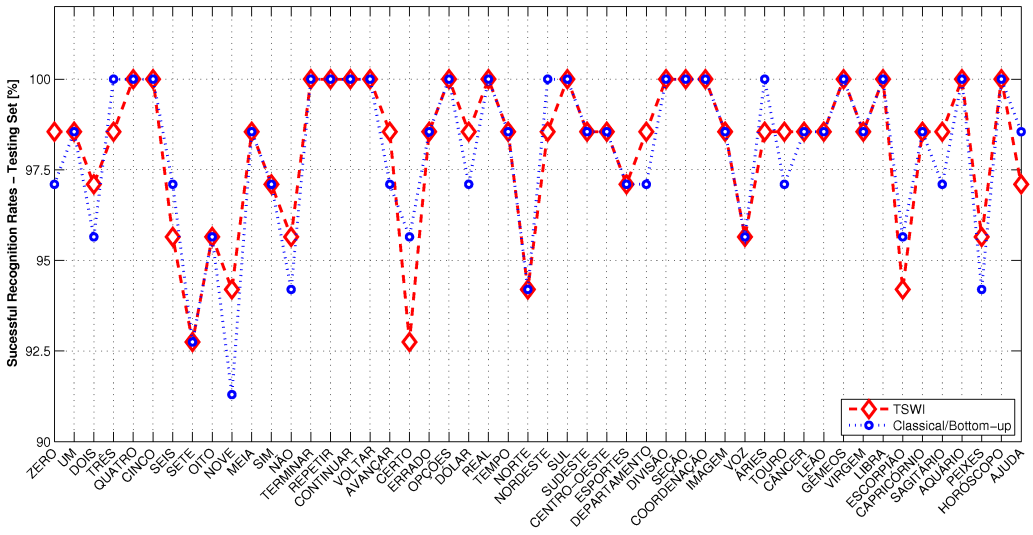



Figure 5.2: Comparison of the testing set successful recognition rates, in %, for MFCC-MLP-recognizer using TSWS, Classical and Bottom-up methods.

## 5.3 Smaller Vocabulary

As McLoughlin and Sharifzadeh have stated [44], "we can improve recognition firstly by restricting vocabulary size, and secondly by improving signal-to-noise ratio". As an interesting and easy way to better explore the differences on recognition rates generated by different supporting speech segmentation methods, we have restricted vocabulary size while in the same SNR scenario.

In order to enable the implementation of a menu navigator application, we came with the suggestion of using only 17 words from the original project database, the ones presented in Table 5.4. Training set is now constituted of 2,346 patterns extracted from audio files (first and second utterances) and testing set is constituted of 1,173 patterns extracted from audio files (third utterance).

Table 5.4: Portuguese voice commands from the smaller vocabulary.

| ID | Command | English | IPA |
|---|---|---|---|
| 0 | ZERO | ZERO | /ˈzɛrʊ/ |
| 1 | UM | ONE | /ˈũ/ |
| 2 | DOIS | TWO | /ˈdoyʒ/ |
| 3 | TRÊS | THREE | /ˈtreʒ/ |
| 4 | QUATRO | FOUR | /ˈkwatrʊ/ |
| 5 | CINCO | FIVE | /ˈsĩkʊ/ |
| 6 | SEIS | SIX | /ˈseyʒ/ |
| 7 | SETE | SEVEN | /ˈsɛtɪ/ |
| 8 | OITO | EIGHT | /ˈoytʊ/ |
| 9 | NOVE | NINE | /ˈnɷvɪ/ |
| 10 | MEIA | HALF / SIX | /ˈmeya/ |
| 11 | SIM | YES | /ˈsĩ/ |
| 12 | NÃO | NO | /ˈnãw/ |
| 16 | VOLTAR | BACK | /volˈtaʀ/ |
| 17 | AVANÇAR | FORWARD | /avãˈsaʀ/ |
| 20 | OPÇÕES | OPTIONS | /opˈsõyʒ/ |
| 49 | AJUDA | HELP | /aˈjuda/ |

Here, because of the smaller vocabulary, the contribution of the TSWS method for the recognition system has shown to be more expressive. In Section 5.3.1 can be seen the confusion matrices whose overall ratings are presented in Table 5.5.

Table 5.5: Overall successful recognition rates in % for MFCC-MLP-recognizer with a vocabulary of 17 words.

| **Classical** | | **Bottom-up** | | **TSWS** | |
|---|---|---|---|---|---|
| **Train** | **Test** | **Train** | **Test** | **Train** | **Test** |
| 100.0 | 98.6 | 100.0 | 98.6 | 100.0 | 99.0 |

In this case, the TSWS method has supported MFCC-MLP recognizer to achieve 99.0% of overall successful recognition rates, in robusteness tests (testing set), besides the learning successful rate of 100.0% (using training set). Bottom-up method also has supported MFCC-MLP recognizer to achieve 100% with the training set, but achieved 98.6% on testing set. At this far on the scale, too close to 100.0% of succesful recognition rates, an achieved error reduction of 28.6% (0.4% of improvement on successful rates)is considered an important achievement.

Figure 5.3 shows individual rates for each one of the 17 commands from the conducted tests. Only testing set results are here represented, because training set has achieved 100.0% of successful performance for all compared methods.

Another set of tests were conducted, this time with the artificial addition of white Gaussian noise to reach input audio signals with a SNR of 15dB. Table 5.6 presents the overall successful recognition rates achieved, using both Classical, Bottom-up and TSWS methods. Related confusion matrices can be found in Section 5.3.2.

As one can see, the TSWS method had significant contributions to the improvement of the recognition system, even in the presence of artificially added

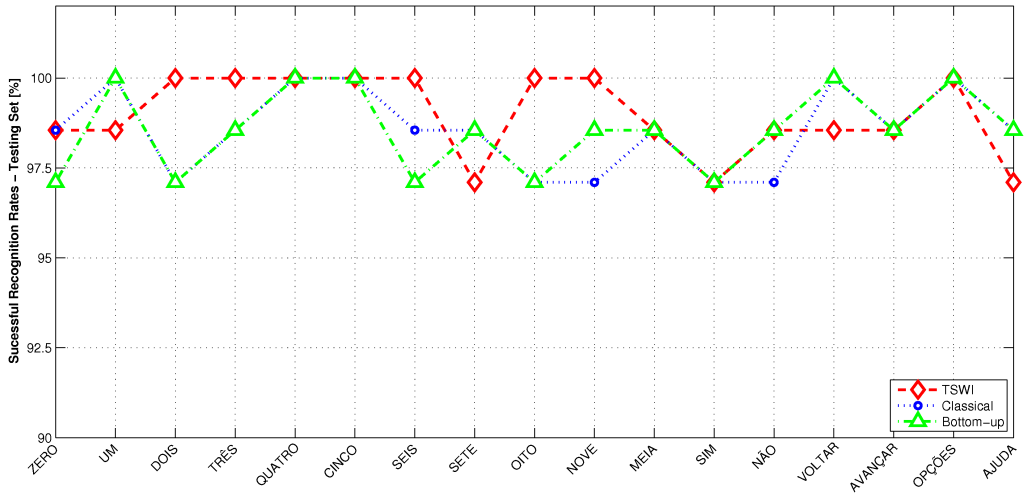



Figure 5.3: Comparison of the testing set successful recognition rates in % for MFCC-MLP-recognizer using TSWS, Classical and Bottom-up methods with smaller vocabulary.

Table 5.6: Overall successful recognition rates in % for MFCC-MLP-recognizer with a vocabulary of 17 words and SNR of 15dB.

| **Classical** | | **Bottom-up** | | **TSWS** | |
|---|---|---|---|---|---|
| **Train** | **Test** | **Train** | **Test** | **Train** | **Test** |
| 99.8 | 93.6 | 99.9 | 91.4 | 100.0 | 96.5 |

noise. A difference of more than 5.0% between successful recognition rates (an error reduction of 59.3%) on nontrained patterns could be evaluated from the performance of the MFCC-MLP-recognizer when it is supported by TSWS method, instead of when supported by Bottom-up method. When comparing Classical and TSWS methods, we have achieved an increase of almost 3.0% (an error reduction of 45.3%) in the performance on nontrained patterns successful recognition. Learning capability was also increased, achieving 100.0% only when the recognizer is supported by the TSWS method. Figures 5.4 and 5.5 show individual rates for each one of the 17 commands from the conducted tests.

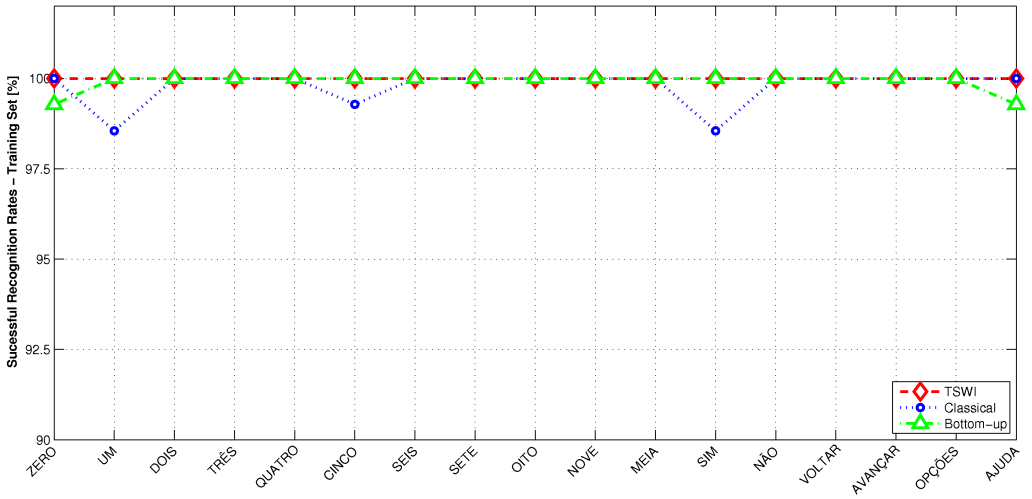



Figure 5.4: Comparison of the training set successful recognition rates in % for MFCC-MLP-recognizer using TSWS, Classical and Bottom-up methods, with addition of WGN to achieve a SNR of 15dB.

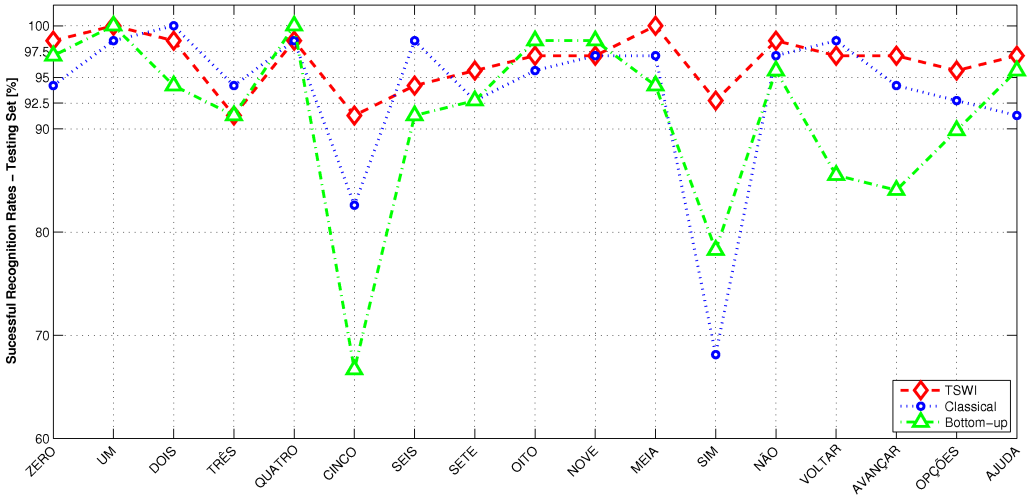



Figure 5.5: Comparison of the testing set successful recognition rates in % for MFCC-MLP-recognizer using TSWS, Classical and Bottom-up methods, with addition of WGN to achieve a SNR of 15dB.

### 5.3.1 Confusion Matrices for 17 Words Vocabulary

Here we present the confusion matrices generated for this project recognition system. The recognition system has *Mel-frequency Cepstral Coefficients* as the parametric representation of the speech signal, and is constituted by a standard multilayer feed-forward network (MLP). The *training set* for the MLP is composed of 2,346 audio signals and pretends to evaluate the recognizer's learning capability. The *testing set* is composed of 1,173 audio signals and pretends to evaluate the generalization ability of the adopted recognizer. Results presented assume the TSWS, Classical or Bottom-up methods, one at a time, to the spoken word segmentation stage.

### TSWS method

Table 5.7: Confusion matrix for *training set* using TSWS and *Mel-Cepstral Coefficients* [Clear signals] with rates in %. Overall recognition rate of 100.0%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SETE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** |

Table 5.8: Confusion matrix for *testing set* using TSWS and *Mel-Cepstral Coefficients* [Clear signals] with rates in %. Overall recognition rate of 99.0%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 01.4 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 01.4 | 00.0 | 00.0 | 01.4 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 |
| SETE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 01.4 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 01.4 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** |

### Classical method

Table 5.9: Confusion matrix for *training set* using *Classical* and *Mel-Cepstral Coefficients* [Clear signals] with rates in %. Overall recognition rate of 100.0%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SETE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** |

Table 5.10: Confusion matrix for *testing set* using *Classical* and *Mel-Cepstral Coefficients* [Clear signals] with rates in %. Overall recognition rate of 98.6%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 |
| SETE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** |

### Bottom-up method

Table 5.11: Confusion matrix for *training set* using *Bottom-up* and *Mel-Cepstral Coefficients* [Clear signals] with rates in %. Overall recognition rate of 100.0%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SETE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** |

Table 5.12: Confusion matrix for *testing set* using *Bottom-up* and *Mel-Cepstral Coefficients* [Clear signals] with rates in %. Overall recognition rate of 98.6%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 |
| SETE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** |

### 5.3.2 Confusion Matrices for 17 Words Vocabulary with SNR 15dB

Here we present the confusion matrices generated for this project recognition system when we artificially add white Gaussian noise to reach a SNR of 15 dB. Again, the *training set* for the MLP is composed of 2,346 audio signals and the *testing set* is composed of 1,173 audio signals. Results presented assume the TSWS, Classical or Bottom-up methods, one at a time, to the spoken word segmentation stage.

### TSWS method

Table 5.13: Confusion matrix for *training set* using TSWS and *Mel-Cepstral Coefficients* [SNR 15dB] with rates in %. Overall recognition rate of 100.0%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SETE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** |

Table 5.14: Confusion matrix for *testing set* using TSWS and *Mel-Cepstral Coefficients* [SNR 15dB] with rates in %. Overall recognition rate of 96.5%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 02.9 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 02.9 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 |
| TRÊS | 00.0 | 00.0 | 01.4 | **91.3** | 00.0 | 00.0 | 04.3 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **91.3** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 05.8 | 00.0 | 00.0 | **94.2** | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SETE | 01.4 | 00.0 | 00.0 | 02.9 | 00.0 | 00.0 | 00.0 | **95.7** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 01.4 | 01.4 | 00.0 | 01.4 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 08.7 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | **92.8** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 02.9 | 01.4 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **95.7** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** |

### Classical method

Table 5.15: Confusion matrix for *training set* using *Classical* and *Mel-Cepstral Coefficients* [SNR 15dB] with rates in %. Overall recognition rate of 99.8%.

| OUTPUT CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **99.3** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.7 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SETE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.7 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.7 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** |
| | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |

TARGET CLASS

Table 5.16: Confusion matrix for *testing set* using *Classical* and *Mel-Cepstral Coefficients* [SNR 15dB] with rates in %. Overall recognition rate of 93.6%.

| OUTPUT CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **94.2** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 05.8 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 |
| DOIS | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 02.9 | 00.0 | 00.0 | 00.0 | 02.9 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **94.2** | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | **82.6** | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 29.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 02.9 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 |
| SETE | 04.3 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **92.8** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **95.7** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 |
| SIM | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 17.4 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | **68.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | **98.6** | 02.9 | 02.9 | 02.9 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 01.4 | 00.0 | 00.0 | **94.2** | 01.4 | 02.9 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | **92.8** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **91.3** |
| | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |

TARGET CLASS

### Bottom-up method

Table 5.17: Confusion matrix for *training set* using *Bottom-up* and *Mel-Cepstral Coefficients* [SNR 15dB] with rates in %. Overall recognition rate of 99.9%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **99.3** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| DOIS | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SETE | 00.7 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.7 |
| AVANÇAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 |
| AJUDA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **99.3** |

Table 5.18: Confusion matrix for *testing set* using *Bottom-up* and *Mel-Cepstral Coefficients* [SNR 15dB] with rates in %. Overall recognition rate of 91.4%.

| OUTPUT CLASS \ TARGET CLASS | ZERO | UM | DOIS | TRÊS | QUATRO | CINCO | SEIS | SETE | OITO | NOVE | MEIA | SIM | NÃO | VOLTAR | AVANÇAR | OPÇÕES | AJUDA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ZERO | **97.1** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 05.8 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 |
| UM | 00.0 | **100** | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 02.9 | 01.4 | 00.0 | 02.9 | 00.0 |
| DOIS | 00.0 | 00.0 | **94.2** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 04.3 | 00.0 |
| TRÊS | 00.0 | 00.0 | 00.0 | **91.3** | 00.0 | 01.4 | 04.3 | 00.0 | 00.0 | 01.4 | 04.3 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| QUATRO | 00.0 | 00.0 | 00.0 | 00.0 | **100** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 |
| CINCO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **66.7** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 20.3 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SEIS | 00.0 | 00.0 | 00.0 | 08.7 | 00.0 | 00.0 | **91.3** | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SETE | 02.9 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **92.8** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 |
| OITO | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| NOVE | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **98.6** | 00.0 | 00.0 | 00.0 | 01.4 | 01.4 | 00.0 | 00.0 |
| MEIA | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | **94.2** | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 |
| SIM | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 29.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | **78.3** | 00.0 | 00.0 | 00.0 | 01.4 | 00.0 |
| NÃO | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **95.7** | 00.0 | 00.0 | 00.0 | 00.0 |
| VOLTAR | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **85.5** | 13.0 | 00.0 | 01.4 |
| AVANÇAR | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 04.3 | **84.1** | 01.4 | 02.9 |
| OPÇÕES | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 02.9 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 01.4 | 02.9 | 01.4 | **89.9** | 00.0 |
| AJUDA | 00.0 | 00.0 | 01.4 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | 00.0 | **95.7** |

# Chapter 6

# Conclusion

In this work, a novel method for speech segmentation and a speech recognition system based on Mel-frequency cepstral coefficients is presented.

The speech segmentation method, named "TEO-based method for Spoken Word Segmentation" (TSWS), has proved to be more efficient than the Classical method, based on energy and zero-crossing rate computations, or even the Bottom-up method, based on concepts of adaptive level equalization, energy pulse detection, and ordering of found boundary. As an example, we can detach the case of the English word "CHIN" /tʃɪn/ presented in Figure 4.4, when both Classical and Bottom-up methods missed the nasal /n/. Considering the manual speech segmentation as the reference, comparisons were performed in order to evaluate the error on endpoints detection for the mentioned methods. The improvement on precision reduced the overall RMSE of 67.8% when compared to the Classical method, and of 61.2%, when compared to the Bottom-up method. Even when dealing with noisy versions of original audio signals, TSWS had also a very interesting performance when compared to Classical and Bottom-up methods (Table 4.3). The error reduc-

tion was of 27.1% for comparison with Classical method, and of 48.5% for comparison with Bottom-up method.

The speaker-independent speech recognition system presented here for isolated words from a limited vocabulary, supported by the proposed speech segmentation method, has achieved excellent recognition rates — 99.0% on average for the generalization of the smaller vocabulary case, against 98.6% of other two comparison methods (a 28.6% reduction on error rate). It also has presented a good generalization performance when dealing with noisy versions of the audio signals that constituted the smaller vocabulary case — achieving 96.5% on average of generalization successful rates when dealing with white Gaussian noise artificially added to audio signals (SNR of 15dB), against 93.6% when using Classical method (45.3% reduction on error rate) and 91.4% for Bottom-up method (59.3% reduction on error rate). Note that, for training sets in the SNR 15dB experiment, the TSWS was the only method which enabled the recognizer to achieve 100.0% of learning capability.

Facing the fact that there is no ultimate solution for all possible cases, this work presents a very adaptable solution for isolated words recognition problems. Solutions presented here have competitive computational costs and point to be very efficient even in low SNR conditions. Results also point to a relatively robust complete solution for several possible applications, including microcontroller-based systems. Other tests must be conducted, though, this time with a more diversified database.

## 6.1 Main Contribution

The main contribution of this work is the proposed TSWS method. With a competitive computational cost, even integrating the preprocessing stage, it

could be easily applied to several applications, specially those with real time requirements. The TSWS algorithm is ready to be integrated to the audio capture system and can be easily converted to speech detection applications.

## 6.2 Ongoing Work

Several paths to the continuation of this research are proposed.

### Automatic evaluation for SNR-dependent constant

The automatic evaluation for SNR-dependent constant of the TSWS method is an important achievement to be made. In fact, it means that the reference value used to the decision if a frame has speech information or not, depends on this constant. Two paths are possible: one, to find a way to the automatic evaluation of this constant; two, to study stochastic noise probability distributions to find out another reliable way to achieve this reference value, depending on the current audio signal samples.

### Database construction of phonetic sounds of Brazilian Portuguese

Following the example of the support database here used, kindly provided by Compton [10], there is the need for constructing a database constituted of nearly all phonetic sounds of Brazilian Portuguese. This kind of database would be very useful when studying the impact of different phonemes to the speech segmentation methods, or even to the recognition system.

### Database construction of Brazilian Portuguese voice commands

There is the need for constructing a more complete isolated word database for Brazilian Portuguese. In fact, the necessary vocabulary for several different applications must be observed. Also, different utterances and accents must be included into this updated database. Noisy patterns, captured in a way to respect Lombard reflex, are also planned to be included in this novel database.

### Other human auditory models

When one comes to real time applications, some aspects are uncovered. Recognizer's reliability is very important when migrating from recorded patterns from a database to real word applications. Some other human auditory models and other topologies for the recognition stage must be investigated in other to increase reliability of real time speech recognition systems.

## 6.3 Publications

### List of Publications

- Peretta, I. S.; Lima, G. F. M.; Tavares, J. A. & Yamanaka, K. Reconhecimento de Comando de Voz Baseado em Filtros Wavelet Utilizando Redes Neurais Artificiais. In: *Anais do IX Congresso Brasileiro de Redes Neurais e Inteligência Computacional (CD-Rom)*, 2009.

- Peretta, I. S.; Lima, G. F. M.; Tavares, J. A. & Yamanaka, K. A Spoken Word Boundaries Detection Strategy for Voice Command Recognition. *Learning & Nonlinear Models*, vol. 8, no. 3, 2010.

[Online] journal available at http://www.deti.ufc.br/~lnlm/index.php?v=8&n=3.

### Database Online Publishing

We made the files from the support database, used by this work in Chapter 4, available for future works on speech segmentation methods, as well as the C++ Class source code for the TSWS method, presented in Appendix B. One can download any of them from: http://speechsegmentbm.sourceforge.net/

# Bibliography

[1] ETSI Standard for Speech Processing, Transmission and Quality aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms.ETSI Standard for Speech Processing, Transmission and Quality aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms., September 2003. [Online] available at `http://webapp.etsi.org/workprogram/Report_WorkItem.asp?WKI_ID=18820`.

[2] Aertsen, A. M. H. J. & Johannesma, P. I. M. Spectro-Temporal Receptive Fields of Auditory Neurons in the Grassfrog. *Biological Cybernetics*, vol. 38, no. 4, 1980.

[3] Atal, B. S. & Hanauer, S. L. Speech Analysis and Synthesis by Linear Prediction of the Speech Wave. *The Journal of the Acoustical Society of America*, vol. 50, no. 28, 1971.

[4] Bai, J. & Zhang, X. Research of a Non-Specific Person Noise-Robust Speech Recognition System. In: *Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference on*, September 2009, p. 1–4.

[5] Bo, L.; Dong-xia, W.; De-jun, Z. & Tie-sen, H. On speech recognition access control system based on HMM/ANN. In: *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, July 2010, vol. 2, p. 682–686.

[6] Bourlard, H. & Morgan, N. Hybrid HMM/ANN systems for speech recognition: Overview and new research directions. In: *Adaptive Processing of Sequences and Data Structures*, C. Giles & M. Gori, Eds., vol. 1387 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 1998.

[7] Burges, C. J. C. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, vol. 2, no. 2, June 1998.

[8] Carrasquillo, P. A. T.; Singer, E.; Kohler, M. A. & Deller, J. R. Approaches to language identification using Gaussian mixture models and shifted delta cepstral features. In: *Proc. ICSLP 2002*, 2002, p. 89–92.

[9] Combrinck, H. P. & Botha, E. On the Mel-scaled Cepstrum, 1996. [Online] available at `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.18.1382&rep=rep1&type=pdf`.

[10] Compton, A. J. Phonetic Symbols Table: Phonetic Symbols for the Consonant and Vowel Sounds of American English, March 2010. accessed on oct 17th, 2010 at 11am [Online] available at `http://comptonpeslonline.com/phonetic_symbols_table.shtml`.

[11] Cortes, C. & Vapnik, V. Support-vector networks. *Machine Learning*, vol. 20, no. 3, 1995.

[12] Csáji, B. C. Approximation with Artificial Neural Networks. Master's Thesis, Faculty of Science, Eötvös Loránd University, 2001. [Online] available at `http://www.sztaki.hu/~csaji/CsBCs_MSc.pdf`.

[13] Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, vol. 2, no. 4, 1989.

[14] Daubechies, I. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, 1992.

[15] Davis, K. H.; Biddulph, R. & Balashek, S. Automatic Recognition of Spoken Digits. *The Journal of the Acoustical Society of America*, vol. 24, no., November 1952.

[16] Davis, S. B. & Mermelstein, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 4, August 1980.

[17] Fausett, L. *Fundamentals of Neural Networks*. Prentice Hall, December 1993.

[18] Fritsch, F. N. & Carlson, R. E. Monotone Piecewise Cubic Interpolation. *SIAM Journal on Numerical Analysis*, vol. 17, no. 2, April 1980. [Online] available at `http://www.jstor.org/stable/2156610`.

[19] Gandhiraj, R. & Sathidevi, P. S. Auditory-Based Wavelet Packet Filterbank for Speech Recognition Using Neural Network. In: *Advanced Computing and Communications, 2007. ADCOM 2007. International Conference on*, December 2007, p. 666–673.

[20] Ghaemmaghami, H.; Vogt, R.; Sridharan, S. & Mason, M. Speech Endpoint Detection Using Gradient Based Edge Detection Techniques. In: *Signal Processing and Communication Systems, 2008. ICSPCS 2008. 2nd International Conference on*, 2008, vol. December, p. 1–8.

[21] Glasberg, B. R. & Moore, B. C. J. Derivation of auditory filter shapes from notched-noise data. *Hearing Research*, vol. 47, no. 1-2, 1990.

[22] Greenwood, D. D. Critical Bandwidth and the Frequency Coordinates of the Basilar Membrane. *The Journal of the Acoustical Society of America*, vol. 33, no. 10, October 1961.

[23] Haykin, S. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, second ed., July 1998.

[24] Hochberg, M. M.; Renals, S. J.; Robinson, A. J. & Cook, G. D. Recent improvements to the ABBOT large vocabulary CSR system. In: *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, May 1995, vol. 1, p. 69–72.

[25] Hornik, K. Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks*, vol. 4, no., March 1991.

[26] Huda, M. N.; Hasan, M. M.; Ahmed, S.; Rahman, D. F.; Muhammad, G.; Kotwal, M. R. A.; Banik, M. & Hossain, M. S. Distinctive Phonetic Features (DPFs)-Based Isolated Word Recognition Using Multilayer Neural Networks. In: *Integrated Intelligent Computing (ICIIC), 2010 First International Conference on*, August 2010, p. 51 –55.

[27] Itakura, F. Minimum prediction residual principle applied to speech recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 23, no. 1, 1975.

[28] Junqua, J.-C. The influence of acoustics on speech production: A noise-induced stress phenomenon known as the Lombard reflex. *Speech Communication*, vol. 20, no. 1-2, 1996.

[29] Kaiser, J. F. On a simple algorithm to calculate the 'energy' of a signal. In: *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, April 1990, vol. 1, p. 381–384.

[30] Kaiser, J. F. Some useful properties of Teager's energy operators. In: *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, April 1993, vol. 3, p. 149–152.

[31] Ketabdar, H. & Bourlard, H. Enhanced Phone Posteriors for Improving Speech Recognition Systems. *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 18, no. 6, August 2010.

[32] Kim, H. K. & Rose, R. C. Cepstrum-domain model combination based on decomposition of speech and noise for noisy speech recognition. In: *Acoustics, Speech, and Signal Processing, 2002. Proceedings. (ICASSP '02). IEEE International Conference on*, 2002, vol. 1, p. I.209–I.212.

[33] Kriesel, D. *A Brief Introduction to Neural Networks*. 2007. [Online] available at `http://www.dkriesel.com`.

[34] Kurian, C. & Balakrishnan, K. Speech recognition of Malayalam numbers. In: *Nature Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, 2009.

[35] Lamel, L.; Rabiner, L.; Rosenberg, A. & Wilpon, J. An improved endpoint detector for isolated word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 29, no. 4, 1981.

[36] Liu, Y.; Shriberg, E.; Stolcke, A.; Hillard, D.; Ostendorf, M. & Harper, M. Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 14, no. 14, September 2006.

[37] Lyon, R. F. *Neuromorphic systems engineering*, chapt. Filter cascades as analogs of the cochlea, p. 3–18. Kluwer Academic, 1998.

[38] Lyon, R. F.; Katsiamis, A. G. & Drakakis, E. M. History and Future of Auditory Filter Models. In: *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, May 2010, p. 3809–3812.

[39] Maesschalck, R. D.; Jouan-Rimbaud, D. & Massart, D. L. The Mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 1, 2000.

[40] Maorui, B.; Mingming, F. & Yuzheng, Z. Speech Recognition System Using a Wavelet Packet and Synergetic Neural Network. In: *Measuring Technology and Mechatronics Automation (ICMTMA), 2010 International Conference on*, March 2010, vol. 3, p. 453–456.

[41] Maragos, P.; Quatieri, T. F. & Kaiser, J. F. Detecting Nonlinearities in Speech using an Energy Operator. In: *Digital Signal Processing, Proceedings of 1990 IEEE International Workshop on*, September 1990, p. 1–2.

[42] Martins, J. A. *Avaliação de Diferentes Técnicas para Reconhecimento de Fala*. PhD Thesis, Universidade Estadual de Campinas — UNICAMP, December 1997.

[43] McCulloch, W. & Pitts, W. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biology*, vol. 5, no. 4, December 1943.

[44] McLoughlin, I. & Sharifzadeh, H. R. *Speech Recognition Technologies and Applications*, chapt. Speech Recognition for Smart Homes, p. 477–494. In-Teh, November 2008.

[45] Mermelstein, P. Automatic segmentation of speech into syllabic units. *The Journal of the Acoustical Society of America*, vol. 58, no. 4, 1975.

[46] Møller, M. F. A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. *Neural Networks*, vol. 6, no. 4, 1993.

[47] Mporas, I.; Ganchev, T. & Fakotakis, N. Speech segmentation using regression fusion of boundary predictions. *Computer Speech & Language*, vol. 24, no. 2, 2010.

[48] Oppenheim, A. V. & Schafer, R. W. From frequency to quefrency: a history of the cepstrum. *Signal Processing Magazine, IEEE*, vol. 21, no. 21, September 2004.

[49] O'Shaughnessy, D. *Speech communication: human and machine*. Addison-Wesley Pub. Co., 1987.

[50] Patterson, R. D. & Holdsworth, J. *Advances in Speech, Hearing and Language Processing*, vol. 3, Part B, chapt. A Functional

Model of Neural Activity Patterns and Auditory Images, p. 547–558. JAI Press, 1996. remastered version for electronic distribution [Online] available at `http://www.pdn.cam.ac.uk/groups/cnbh/research/publications/pdfs/PH96.pdf`.

[51] Paul, A.; Das, D. & Kamal, M. Bangla Speech Recognition System Using LPC and ANN. In: *Advances in Pattern Recognition, 2009. ICAPR '09. Seventh International Conference on*, February 2009, p. 171–174.

[52] Peretta, I. S.; Lima, G. F. M.; Tavares, J. A. & Yamanaka, K. Reconhecimento de Comando de Voz Baseado em Filtros Wavelet Utilizando Redes Neurais Artificiais. In: *Anais do IX Congresso Brasileiro de Redes Neurais e Inteligência Computacional*, October 2009. electronic distribution.

[53] Peretta, I. S.; Lima, G. F. M.; Tavares, J. A. & Yamanaka, K. A Spoken Word Boundaries Detection Strategy for Voice Command Recognition. *Learning & Nonlinear Models*, vol. 8, no. 3, 2010. [Online] journal available at `http://www.deti.ufc.br/~lnlm/index.php?v=8&n=3`.

[54] Rabiner, L.; Levinson, S.; Rosenberg, A. & Wilpon, J. Speaker independent recognition of isolated words using clustering techniques. In: *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '79.*, April 1979, vol. 4, p. 574–577.

[55] Rabiner, L. R. A tutorial on hidden Markov models and selected applications in speech recognition. In: *Readings in speech recognition*, A. Waibel & K.-F. Lee, Eds. Morgan Kaufmann, 1990.

[56] Rabiner, L. R. & Sambur, M. R. Algorithm for determining the endpoints of isolated utterances. *The Journal of the Acoustical Society of America*, vol. 56, no. S1, November 1974.

[57] Rigoll, G. & Neukirchen, C. A New Approach to Hybrid HMM/ANN Speech Recognition Using Mutual Information Neural Networks. In: *Advances in Neural Information Processing Systems 9, NIPS*96*, 1996, The MIT Press, p. 772–778.

[58] Rosenblatt, F. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, vol. 65, no. 6, November 1958.

[59] Rumelhart, D. E.; Hinton, G. E. & Williams, R. J. *Parallel distributed processing: explorations in the microstructure of cognition*, vol. 1, chapt. Learning internal representations by error propagation, p. 318–362. MIT Press, 1986.

[60] Schluter, R.; Bezrukov, L.; Wagner, H. & Ney, H. Gammatone Features and Feature Combination for Large Vocabulary Speech Recognition. In: *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, April 2007, vol. 4, p. IV.649 – IV.652.

[61] Shannon, C. E. Communication in the Presence of Noise. *Proceedings of the IEEE*, vol. 86, no. 2, February 1998.

[62] Shao, Y.; Jin, Z.; Wang, D. & Srinivasan, S. An auditory-based feature for robust speech recognition. In: *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, April 2009, p. 4625–4628.

[63] Shao, Y. & Wang, D. Robust speaker identification using auditory features and computational auditory scene analysis. In: *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, April 2008, p. 1589–1592.

[64] Stevens, S. S.; Volkmann, J. & Newman, E. B. A Scale for the Measurement of the Psychological Magnitude Pitch. *The Journal of the Acoustical Society of America*, vol. 8, no. 3, 1937.

[65] Sukumar, A. R.; Shah, A. F. & Anto, P. B. Isolated question words recognition from speech queries by using Artificial Neural Networks. In: *Computing Communication and Networking Technologies (ICCCNT), 2010 International Conference on*, July 2010, p. 1–4.

[66] Thiang, D. W. Limited Speech Recognition for Controlling Movement of Mobile Robot Implemented on ATmega162 Microcontroller. In: *Computer and Automation Engineering, 2009. ICCAE '09. International Conference on*, March 2009, p. 347–350.

[67] Tian, L. & Noore, A. *Computational Intelligence in Reliability Engineering*, vol. 39 of *Studies in Computational Intelligence*, chapt. Computational Intelligence Methods in Software Reliability Prediction, p. 375–397. Springer, 2007.

[68] Vaidyanathan, P. P. *The Theory of Linear Prediction*. Synthesis Lectures on Signal Processing #3. Morgan & Claypool, 2008. ebook.

[69] Vidal, E. & Marzal, A. A review and new approaches for automatic segmentation of speech signals. In: *Signal Processing V: Theories and Applications*, L. Torres, E. Masgrau, & M. A. Lagunas, Eds. Elsevier Science Publishers B.V., 1990.

[70] Wong, E. & Sridharan, S. Comparison of linear prediction cepstrum coefficients and mel-frequency cepstrum coefficients for language identification. In: *Intelligent Multimedia, Video and Speech Processing, 2001. Proceedings of 2001 International Symposium on*, 2001, p. 95–98.

[71] Yi, L. & Yingle, F. A Novel Algorithm to Robust Speech Endpoint Detection in Noisy Environments. In: *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, May 2007, p. 1555–1558.

[72] Zhang, W. Q.; He, L.; Deng, Y.; Liu, J. & Johnson, M. T. Time-Frequency Cepstral Features and Heteroscedastic Linear Discriminant Analysis for Language Recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 2, February 2011. date of current version October 27, 2010.

[73] Zhou, M. Z. & Ji, L. X. Real-Time Endpoint Detection Algorithm Combining Time-Frequency Domain. In: *Intelligent Systems and Applications (ISA), 2010 2nd International Workshop on*, May 2010, p. 1–4.

[74] Zue, V. & Cole, R. *Survey of the State of the Art in Human Language Technology*, chapt. Overview, p. 1–3. Cambridge University Press and Giardini, web ed., 1997. [Online] available at `http://cslu.cse.ogi.edu/HLTsurvey/HLTsurvey.html`.

[75] Zue, V.; Cole, R. & Ward, W. *Survey of the State of the Art in Human Language Technology*, chapt. Speech Recognition, p. 3–10. Cambridge University Press and Giardini, web ed., 1997. [Online] available at `http://cslu.cse.ogi.edu/HLTsurvey/HLTsurvey.html`.

# Appendix A

# Compton's Database

In order to evaluate proposed TSWS segmentation method (see Chapter 4), the following database was used. This database was kindly provided by Arthur J. Compton, Ph.D., from the Compton Phonological Assessment of Foreign Accent, Institute of Language and Phonology [10]. All phonetic symbols included in this database represent nearly all consonant and vowel sounds of American English.

Table A.1: Compton's database audio files with respective phonetic symbols and recorded words.

| ID | Audio file | IPA symbol | Words |
|---|---|---|---|
| | | *Affricates (Unvoiced)* | |
| 1 | audio_t_F.mp3 | /tʃ/ | chin, champ, chew, rich, match, watch |

*continued on next page*

*continued from previous page*

| ID | Audio file | IPA symbol | Words |
|---|---|---|---|
| | | *Affricates (Voiced)* | |
| 2 | audio_d_Z.mp3 | /dʒ/ | jelly, jam, juice, bridge, badge, wage |
| | | *Diphthongs* | |
| 3 | audio_a_6.mp3 | /aɚ/ | are, far, hard |
| 4 | audio_a_i.mp3 | /ai/ | I, my, like |
| 5 | audio_a_U.mp3 | /aʊ/ | cow, out, down |
| 6 | audio_e_6.mp3 | /eɚ/ | air, fair, wear |
| 7 | audio_i_6.mp3 | /iɚ/ | ear, fear, beer |
| 8 | audio_o_6.mp3 | /oɚ/ | or, cord, sore |
| 9 | audio_o_i.mp3 | /oi/ | boy, voice, toy |
| | | *Frictions (Unvoiced)* | |
| 10 | audio_F.mp3 | /ʃ/ | ship, shall, shoe, wish, cash, push |
| 11 | audio_f.mp3 | /f/ | feet, fat, foot, if, off, enough |
| 12 | audio_O.mp3 | /θ/ | think, thumb, thought, fifth, bath, mouth |
| 13 | audio_s.mp3 | /s/ | see, sat, Sue, kiss, gas, juice |
| | | *Frictions (Voiced)* | |
| 14 | audio_H.mp3 | /ð/ | this, that, those, breathe, bathe, smooth |
| 15 | audio_v.mp3 | /v/ | very, vat, voice, leave, of, move |
| 16 | audio_Z.mp3 | /ʒ/ | beige, rouge, vision |
| 17 | audio_z.mp3 | /z/ | zip, zone, zoo, is, jazz, froze |

*continued on next page*

*continued from previous page*

| ID | Audio file | IPA symbol | Words |
|---|---|---|---|
| | | *Glides* | |
| 18 | audio_h.mp3 | /h/ | he, hat, who |
| 19 | audio_j.mp3 | /j/ | yes, young, youth |
| 20 | audio_w.mp3 | /w/ | we, why, wood |
| | | *Liquids* | |
| 21 | audio_l.mp3 | /l/ | lead, lack, look, feel, shall, pull |
| 22 | audio_r.mp3 | /r/ | read, ran, ruby |
| | | *Nasals* | |
| 23 | audio_m.mp3 | /m/ | meet, mat, move, team, ham, come |
| 24 | audio_n.mp3 | /n/ | neat, knack, new, seen, can, soon |
| 25 | audio_N.mp3 | /ŋ/ | wing, sang, tongue |
| | | *Stops (Unvoiced)* | |
| 26 | audio_k.mp3 | /k/ | keep, cap, cook, tick, wake, look |
| 27 | audio_p.mp3 | /p/ | peak, pack, push, keep, cap, soup |
| 28 | audio_t.mp3 | /t/ | tea, tag, took, heat, sat, boot |
| | | *Stops (Voiced)* | |
| 29 | audio_b.mp3 | /b/ | beat, back, bush, rib, cab, robe |

*continued on next page*

*continued from previous page*

| ID | Audio file | IPA symbol | Words |
|---|---|---|---|
| 30 | audio_d.mp3 | /d/ | deep, dad, do, need, pad, wood |
| 31 | audio_g.mp3 | /ɡ/ | give, gas, go, dig, egg, dog |
| | | *Vowels (Back)* | |
| 32 | audio_a.mp3 | /a/ | hot, ball, off |
| 33 | audio_o.mp3 | /o/ | boat, no, sew |
| 34 | audio_U.mp3 | /ʊ/ | took, foot, puss |
| 35 | audio_u.mp3 | /u/ | Sue, boot, moon |
| | | *Vowels (Central)* | |
| 36 | audio_6.mp3 | /ɚ/ | mother, paper, later |
| 37 | audio_7.mp3 | /ʌ/ | but, luck, fun |
| 38 | audio_9.mp3 | /ɝ/ | bird, nurse, learn |
| 39 | audio_C.mp3 | /ə/ | about, upon, alike |
| | | *Vowels (Front)* | |
| 40 | audio_1.mp3 | /æ/ | as, fat, back |
| 41 | audio_E.mp3 | /ɛ/ | wed, step, neck |
| 42 | audio_e.mp3 | /e/ | play, date, take |
| 43 | audio_I.mp3 | /ɪ/ | it, sit, kick |
| 44 | audio_i.mp3 | /i/ | eat, see, me |
| | | *l-Blends* | |
| 45 | audio_b_l.mp3 | /bl/ | bliss, black, blue |
| 46 | audio_f_l.mp3 | /fl/ | flee, flap, flew |
| 47 | audio_g_l.mp3 | /ɡl/ | glide, glass, gloom |
| 48 | audio_k_l.mp3 | /kl/ | clean, class, clue |

*continued on next page*

*continued from previous page*

| ID | Audio file | IPA symbol | Words |
|---|---|---|---|
| 49 | audio_p_l.mp3 | /pl/ | please, plant, plow |
| | | *r-Blends* | |
| 50 | audio_b_r.mp3 | /br/ | bring, brass, brew |
| 51 | audio_d_r.mp3 | /dr/ | drip, drag, droop |
| 52 | audio_f_r.mp3 | /fr/ | free, frank, fruit |
| 53 | audio_g_r.mp3 | /gr/ | greed, grand, groom |
| 54 | audio_k_r.mp3 | /kr/ | creep, crack, crude |
| 55 | audio_O_r.mp3 | /θr/ | three, thrift, threw |
| 56 | audio_p_r.mp3 | /pr/ | pretty, praise, proof |
| 57 | audio_t_r.mp3 | /tr/ | tree, trap, true |
| | | *s-Blends* | |
| 58 | audio_s_k.mp3 | /sk/ | skid, scan, school |
| 59 | audio_s_l.mp3 | /sl/ | sleep, slap, slow |
| 60 | audio_s_m.mp3 | /sm/ | smell, smash, smooth |
| 61 | audio_s_n.mp3 | /sn/ | sniff, snag, snoop |
| 62 | audio_s_p.mp3 | /sp/ | spell, Spanish, spoon |
| 63 | audio_s_t.mp3 | /st/ | stick, stamp, stoop |
| 64 | audio_s_w.mp3 | /sw/ | sweet, swam, swoop |
| | | *3-Element Blends* | |
| 65 | audio_s_k_r.mp3 | /skr/ | script, scrap, screw |
| 66 | audio_s_p_l.mp3 | /spl/ | split, splash, splatter |
| 67 | audio_s_p_r.mp3 | /spr/ | spring, spread, spray |
| 68 | audio_s_t_r.mp3 | /str/ | strip, strap, strong |

# Appendix B

# TSWS C++ Class: Source Code

This simple C++ class is ready to be used with saved audio files. It implements the TSWS algorithm for the isolation of a spoken word present in a given audio signal, loaded into an array or a vector of double type. Also, it could be an inspiration to convert this source code to fit real time application requirements, where the audio capture is an integrating part of the algorithm.

There are three public constants:

- *startpoint*: the audio vector index of the sample which is starting point of the spoken word;
- *endpoint*: the audio vector index of the sample which is ending point of the spoken word;

- *lengthIW*: the length, in number of samples, of the isolated spoken word.

Also, there are two public methods:

- *getIWs*: getting as the argument the audio signal, as a vector of doubles, or an array of doubles with its length, this method returns the isolated spoken word signal as a vector of doubles;
- *getIWs_a*: getting as the arguments the audio signal, as an array of doubles, and its length, this method returns the isolated spoken word signal as an array of doubles;

To declare a variable from this class, one should use the class constructor, choosing as arguments:

- $Fs$: the sampling frequency;
- $A$: the SNR-dependent constant (default value = 9);
- $T_{sil}$: time of capture, in ms, in the beginning of the process when the captured signal is considered as silence (default value = 100);
- $T_{frm}$: time of each frame, in ms, to be captured after (default value = 25);
- $T_{min}$: minimum time that an isolated fragment must last to be considered as speech, not interference (default value = 150);
- $T_{saw}$: minimum time of silence after a bound speech to confirm the word signal is fully isolated (default value = 250).

## B.1 Header

File **tsws.h**:

```
#include <vector>
#include <cmath>

class TSWS{
        public:
                TSWS(int Fs, double A = 9.0,
                    int Tsil = 100, int Tfrm = 25,
                    int Tmin = 150, int Tsaw = 250);
                ~TSWS();
                long long startpoint;
                long long endpoint;
                long long lengthIW;
                std::vector<double> getIWs
                    (std::vector<double> s);
                std::vector<double> getIWs(double * s,
                    size_t length);
                double* getIWs_a(double * s, size_t length);
        private:
                std::vector<double> array2vector(double * s,
                    size_t length);
                double* vector2array(std::vector<double> s);
                double maxabs(std::vector<double> x);
                double stdev(std::vector<double> x);
                double mean(std::vector<double> x);
                long long nsil;
                long long nframe;
                long long nmin;
                long long nsaw;
                double A;
};
```

## B.2 Code

File **tsws.cpp**:

```
#include "tsws.h"
```

```
2
3 TSWS::TSWS(int Fs, double SNRconst, int Tsil, int Tfrm,
4         int Tmin, int Tsaw){
5     nsil = (int) (Tsil*Fs/1000);
6     nframe = (int) (Tfrm*Fs/1000);
7     nmin = (int) (Tmin*Fs/1000);
8     nsaw = (int) (Tsaw*Fs/1000);
9     A = SNRconst;
10    startpoint = -1;
11    endpoint = -1;
12 }
13
14 TSWS::~TSWS(){
15 }
16
17 std::vector<double> TSWS::getIWs(std::vector<double> s){
18    std::vector<double> segmentS, sfr, sil, sof, spe,
19       TEO, vosil, aux;
20    double REF, maxTEO;
21    int i, j, a, b, nzeros, saw;
22    bool word;
23    // Silence capture ==========================================
24    if (((s.size() - nsil) % nframe) != 0) {
25       nzeros = nframe - ((s.size() - nsil) % nframe);
26       for (i = 0; i < nzeros; i++)
27           s.push_back(0.0);
28    }
29    sil.clear(); sof.clear(), spe.clear();
30    for (i = 0; i < nsil; i++) sil.push_back(s[i]);
31    //Preprocess ***
32    spe = sof = sil;
33    for (i = 1; i < sil.size(); i++){
34        //offset compensation
35        sof[i] = sil[i] - sil[i-1] + 0.999*sof[i-1];
36        //pre-emphasis
37        spe[i] = sof[i] - 0.97*sof[i-1];
38    }
39    sil = spe;
40    // Update preprocessed silence
41    for (i = 0; i < nsil; i++)
42        s[i] = sil[i];
43    //Apply TEO ***
44    TEO.clear();
45    TEO.push_back(0.0);
```

```
46      for (i = 1; i < sil.size()-1; i++)
47          TEO.push_back(pow(sil[i],2) - sil[i-1]*sil[i+1]);
48      TEO.push_back(0.0);
49      //Reference evaluation ***
50      vosil = TEO;
51      REF = maxabs(vosil) + A*stdev(vosil);
52      // Frames capture ==========================================
53      word = false; saw = 0;
54      for (i = nsil; i < s.size(); i += nframe){
55          a = i; b = i+nframe-1;
56          sfr.clear(); sof.clear(), spe.clear();
57          for (j = a; j <= b; j++) sfr.push_back(s[j]);
58          //Preprocess ***
59          spe = sof = sfr;
60          for (j = 1; j < sfr.size(); j++){
61              //offset compensation
62              sof[j] = sfr[j] - sfr[j-1] + 0.999*sof[j-1];
63              //pre-emphasis
64              spe[j] = sof[j] - 0.97*sof[j-1];
65          }
66          sfr = spe;
67          // Update preprocessed frame
68          for (j = a; j <= b; j++)
69              s[j] = sfr[j];
70          //Apply TEO ***
71          TEO.clear();
72          TEO.push_back(0.0);
73          for (j = 1; j < sfr.size()-1; j++)
74              TEO.push_back(pow(sfr[j],2) -
75                  sfr[j-1]*sfr[j+1]);
76          TEO.push_back(0.0);
77          maxTEO = maxabs(TEO);
78          // Decision if it is speech or not
79          if (word) {
80              if (maxTEO < REF) {
81                  endpoint = b; //set end point
82                  word = false;
83                  if (endpoint - startpoint +1 > nmin) {
84                      saw = 0;
85                  }
86                  else {// remove start and end points
87                      startpoint = -1;
88                      endpoint = -1;
89                  }
```

```
90              }
91          }
92          else {
93              //silence after word update length
94              saw = saw + nframe;
95              if (maxTEO > REF) {
96                  if (startpoint < 0) {
97                      startpoint = a; //set start point
98                      word = true;
99                  }
100                 else {
101                     if (saw <= nsaw){
102                         saw = 0;
103                         word = false;
104                         endpoint = -1;
105                     }
106                 }
107             }
108             else {//Update Reference ***
109                 aux.clear(); aux = vosil; vosil.clear();
110                 for (j = TEO.size(); j < aux.size(); j++)
111                     vosil.push_back(aux[j]);
112                 for (j = 0; j < TEO.size(); j++)
113                     vosil.push_back(TEO[j]);
114                 REF = maxabs(vosil) + A*stdev(vosil);
115                 if (startpoint >= 0 && endpoint >=0 &&
116                     saw > nsaw) break;
117             }
118         }
119     }
120     // If method didn't set endpoint, force to last frame
121     if (startpoint >= 0 && endpoint < 0)
122         endpoint = s.size()-1;
123     // Update isolated word vector
124     segmentS.clear();
125     for (j = startpoint; j <= endpoint; j++)
126         segmentS.push_back(s[j]);
127     lengthIW = segmentS.size();
128     return segmentS;
129 }
130
131 std::vector<double> TSWS::getIWs(double * s,
132         size_t length){
133     std::vector<double> aux = array2vector(s,length);
```

```
134     aux = getIWs(aux);
135     return aux;
136 }
137
138 double* TSWS::getIWs_a(double * s, size_t length){
139     std::vector<double> aux =
140         getIWs(array2vector(s,length));
141     return vector2array(aux);
142 }
143
144 std::vector<double> TSWS::array2vector(double * s,
145         size_t length){
146     std::vector <double> ret;
147     ret.clear();
148     for (size_t i = 0; i < length; i++)
149         ret.push_back(s[i]);
150     return ret;
151 }
152
153 double* TSWS::vector2array(std::vector<double> s){
154     size_t length = s.size();
155     double *ret = new (std::nothrow) double[length];
156     for (size_t i = 0; i < length; i++)
157         ret[i] = s.at(i);
158     return ret;
159 }
160
161 double TSWS::maxabs(std::vector<double> x){
162     std::vector<double>::iterator its;
163     double maxabsX = 0;
164     for (its = x.begin(); its < x.end(); its++)
165         if (std::abs(*its) > maxabsX)
166             maxabsX = std::abs(*its);
167     return maxabsX;
168 }
169
170 double TSWS::stdev(std::vector<double> x){
171     std::vector<double>::iterator its;
172     double meanX = mean(x), summer = 0.0;
173     for (its = x.begin(); its < x.end(); its++)
174         summer += pow(*its - meanX,2);
175     return sqrt(summer/(x.size()-1));
176 }
177
```

```
178 double TSWS::mean(std::vector<double> x){
179     std::vector<double>::iterator its;
180     double summer = 0.0;
181     for (its = x.begin(); its < x.end(); its++)
182         summer += *its;
183     return summer/x.size();
184 }
```

## B.3 Simple Utilization Example

```
1  // ...
2  #include <vector>
3  #include "tsws.h"
4  using namespace std;
5  // ...
6  int main (){
7      // ...
8      // Creates two vector class variables to save the
9      // audio file and the isolated word
10     vector<double> sound, isolword;
11     // Creates a TSWS class variable with Fs = 8kHz
12     // and all parameters set to default
13     TSWS aux(8000);
14     // ...
15     // Get the isolated word vector
16     isolword = aux.getIW(sound);
17     // ...
18     // Another way to get the isolated word vector
19     isolword.clear();
20     for (int i = aux.startpoint; i <= aux.endpoint; i++)
21         isolword.push_back(sound[i]);
22     // ...
23 }
```