UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETROTÉCNICA
CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO

CÉSAR THUMER ANTONELLI

USO DO RECONHECIMENTO DE FALA PARA APLICAÇÃO EM FECHADURA ELETRÔNICA COMANDADA PELA VOZ

CURITIBA
2016

CÉSAR THUMER ANTONELLI

USO DO RECONHECIMENTO DE FALA PARA APLICAÇÃO EM FECHADURA ELETRÔNICA COMANDADA PELA VOZ

Trabalho de Conclusão de Curso de Graduação, apresentado à disciplina de TCC 2, do Curso Superior de Engenharia de Controle e Automação do Departamento Acadêmico de Eletrotécnica – DAELT – da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Dr. Marcelo De Oliveira Rosa

CURITIBA

2016

CESAR THUMER ANTONELLI

Uso do reconhecimento de fala para aplicação em fechadura eletrônica comandada pela voz

Este Trabalho de Conclusão de Curso de Graduação foi julgado e aprovado como requisito parcial para a obtenção do Título de Engenheiro de Controle e Automação, do curso de Engenharia de Controle e Automação do Departamento Acadêmico de Eletrotécnica (DAELT) da Universidade Tecnológica Federal do Paraná (UTFPR).

Curitiba, 09 de agosto de 2016.

_____
Prof. Paulo Sérgio Walenia, Eng.
Coordenador de Curso
Engenharia de Controle e Automação

_____
Prof. Marcelo de Oliveira Rosa, Dr.
Responsável pelos Trabalhos de Conclusão de Curso
de Engenharia de Controle e Automação do DAELT

ORIENTAÇÃO

_____
Prof. Marcelo de Oliveira Rosa, Dr.
Universidade Tecnológica Federal do Paraná
Orientador

BANCA EXAMINADORA

_____
Prof. Marcelo de Oliveira Rosa, Dr.
Universidade Tecnológica Federal do Paraná

_____
Prof. Amauri Amorin Assef, Dr.
Universidade Tecnológica Federal do Paraná

_____
Prof. Antonio Carlos Pinho, Dr.
Universidade Tecnológica Federal do Paraná

_____
Prof. Gustavo Nishida, Dr.
Universidade Tecnológica Federal do Paraná

A folha de aprovação assinada encontra-se na Coordenação do Curso de Engenharia de Controle e Automação.

RESUMO

ANTONELLI; César Thumer. **Uso do reconhecimento da fala para aplicação em fechadura eletrônica comandada pela voz**. 2016. 80 f. Trabalho de Conclusão de Curso (Graduação – Curso de Engenharia de Controle e Automação). Universidade Tecnológica Federal do Paraná, 2016.

Este trabalho tem como objetivo utilizar a tecnologia do reconhecimento de fala para o desenvolvimento de um sistema de reconhecimento de dígitos aplicado em um protótipo de fechadura eletrônica. Como embasamento teórico, são descritos fundamentos fisiológicos da fala e a técnica de extração de parâmetros mel-cepstrais. O sistema de reconhecimento de fala para aplicação em fechadura eletrônica é desenvolvido neste trabalho usando o sistema decodificador de fala Julius em conjunto com o sistema de treinamento HTK, usado para construir e manipular modelos estatísticos denominados Modelos Ocultos de Markov. Em seguida, é descrita a implementação do protótipo com o uso do Raspberry Pi. Os objetivos propostos foram alcançados, com o sistema desenvolvido apresentando um funcionamento satisfatório com diferentes locutores e pode ser validado no protótipo implementado com o Raspberry Pi.

Palavras-Chaves: Reconhecimento de fala, Raspberry Pi, HTK, Decodificador Julius.

ABSTRACT

ANTONELLI; César Thumer. **The use of speech recognition in an application of a voice-based electronic lock**. 2016. 80 f. Trabalho de Conclusão de Curso (Graduação – Curso de Engenharia de Controle e Automação). Universidade Tecnológica Federal do Paraná, 2016.

The aim of the present study is to use the speech recognition technology to the development of a digit recognition system applied to a prototype of an electronic lock. A theoretical background is described the physiological bases of speech and the extraction technique of mel-frequency cepstral parameters. The speech recognition system for using in electronic lockers is developed in this study using the speech decoder system Julius in conjunction with the Hidden Markov Model Toolkit, to build and manipulate statistical models called Hidden Markov Models. Next is described the prototype implementation using the Raspberry Pi. The proposed objectives were achieved, and the developed system presented satisfactory operation with different speakers and could be validated in prototype implemented with the Raspberry Pi.

Keywords: Speech recognition, Raspberry Pi, HTK, Julius Decoder.

LISTA DE FIGURAS

LISTA DE SIGLAS

GPIO	*General Purpose Input/Output*
HMM	*Hidden Markov Model*
HTK	*Hidden Markov Model Toolkit*
ID	Identificação
LED	*Light-Emitting Diode*
MFCC	*Mel Frequency Cepstral Coefficient*
USB	*Universal Serial Bus*
SD	*Secure Digital*
RPi	*Raspberry Pi*
CI	Circuito Integrado
PCM	*Pulse Code Modulation*
FFT	*Fast Fourier Transform*
DCT	*Discrete Fourier Transform*

SUMÁRIO

1 INTRODUÇÃO

1.1 TEMA

Há mais de cinco décadas as tecnologias de reconhecimento de fala vêm sendo desenvolvidas. Atualmente elas fazem parte do cotidiano das pessoas, introduzidas em produtos e serviços.

O processo denominado reconhecimento da fala consiste em mapear o sinal da fala em textos, permitindo seu uso para controlar ações em resposta a comandos falados. Um dos objetivos dessa tecnologia é tornar as tarefas cotidianas mais práticas, proporcionando entre outras vantagens, mais velocidade e produtividade em situações onde o computador é útil às pessoas cujas mãos ou a visão estejam ocupadas de outra forma (SANTOS, 2008).

O reconhecimento de fala também pode ser aplicado para ampliar o acesso de pessoas com deficiências físicas, ajudando-as a executar tarefas do dia-a-dia por meio do comando de voz.

As interfaces com reconhecimento de voz estão rapidamente se tornando uma necessidade e cada vez mais o desenvolvimento de novos sistemas deverá oferecer mais naturalidade na interação entre o homem e a máquina.

1.1.1 Delimitação do Tema

Em reconhecimento de fala, o item locutor refere-se ao individuo que submete sua voz ao processo de reconhecimento. Neste contexto, um modelo acústico é dito ser independente de locutor quando é capaz de reconhecer a fala de uma variedade de locutores, sem a necessidade de qualquer treinamento prévio. Um sistema é denominado de palavras isoladas quando requer que o locutor efetue pausas entre ca-

da palavra falada, já nos sistemas de palavras contínuas, as pausas não são necessárias (DIAS, 2003).

Com base nas classificações acima, pretendeu-se neste trabalho modelar um sistema de reconhecimento de fala discreta com independência de locutor para aplicação em uma fechadura eletrônica por comando de voz. O sistema proposto foi constituído por um vocabulário finito composto especificamente pelos dígitos “0”, “1”, “2”, “3”, “4”, “5”, “6”, “7”, “8” e “9”, pronunciados em idioma Português Brasileiro, que foram usados para definir o código do usuário e a senha de acesso pessoal.

O sistema funciona de forma prática. Primeiro, o usuário deve cadastrar seu código de identificação (ID) e senha de forma manual, com uso do teclado do computador. Para verificação, construiu-se um protótipo que opera por comando de voz. O protótipo foi montado com uso do Raspberry Pi, que é um microcomputador de baixo custo que funciona com diversas distribuições do Linux.

Após efetuar o cadastro, o sistema de reconhecimento foi capaz de validar ID e a senha por meio de comandos falados. No protótipo foram utilizados LEDs *(Light-Emitting Diode)* verde e vermelho para simular a validação do usuário (abertura da fechadura) e a não validação do usuário, respectivamente. Pretendeu-se com o protótipo verificar o desempenho do sistema de reconhecimento de fala modelado.

1.2 PROBLEMAS E PREMISSAS

O ser humano é capaz de interpretar com naturalidade informações complexas provenientes da fala e, dentro dos seus limites, compreender palavras faladas em ambientes ruidosos ou com vários falantes ao mesmo tempo, de forma versátil e sem prejuízos para a comunicação.

Matematicamente um sistema de reconhecimento de fala explora princípios auditivos e fisiológicos do mecanismo de produção da voz. O processo é um desafio, principalmente devido à natureza estocástica do sinal. Para o presente trabalho uma dificuldade é a semelhança acústica existente entre os dígitos e outra são as diferentes pronúncias de locutores distintos (SILVA et al., 2013).

Para este trabalho, é relevante considerar o suporte às pessoas com deficiências físicas. Nesse contexto, a fechadura eletrônica pode ser usada para desenvolvimento de sistemas mais sofisticados de acesso. Permite também que pessoas com deficiências físicas tenham independência e não necessitem de acompanhamento para executar tarefas simples.

Em situações especificas de trabalho como, por exemplo, laboratórios onde deve evitar-se a manipulação de objetos com as mãos, a fechadura eletrônica deverá proporcionar velocidade e outras vantagens.

1.3 OBJETIVOS

1.3.1 Objetivo Geral

O objetivo deste trabalho é modelar um sistema de reconhecimento de fala discreta com independência de locutor para aplicação em fechadura eletrônica usando o sistema decodificador de fala Julius em conjunto com o sistema de treinamento HTK (*Hidden Markov Model Toolkit)*, usado para construir e manipular modelos estatísticos denominados Modelos Ocultos de Markov. Em seguida, implementar um protótipo com o uso do Raspberry Pi para validação do sistema.

1.3.2 Objetivos Específicos

- Compreender os pontos fundamentais do reconhecimento de fala;
- Aprender aspectos teóricos e práticos das ferramentas HTK e Julius;
- Modelar um sistema de reconhecimento de fala;
- Treinar o sistema de reconhecimento para independer de locutor;
- Investigar o desempenho do sistema frente a diferentes locutores;
- Validar o reconhecimento por meio do protótipo usando o Raspberry Pi.

1.4 JUSTIFICATIVA

Pode-se justificar o desenvolvimento deste trabalho a partir de duas perspectivas. Primeiro, o reconhecimento de dígitos falados é um processo fundamental para diversas aplicações via voz. Além de constituírem o vocabulário do sistema proposto, os dígitos são utilizados com frequência em transações bancárias, acesso a banco de dados, compra com cartão de crédito, discagens automáticas, cadastro pessoal, equipamentos industriais onde o reconhecedor poderá ser útil para o operador realizar a entrada de dados em equipamentos sem o uso das mãos ou visão.

Por outra perspectiva, é uma tecnologia que busca oferecer meios que permitam a integração de pessoas com deficiências físicas o que a torna muito interessante. Por essas razões, o tema é sem duvida motivador e fonte de inspiração para pesquisas e novas realizações.

1.5 PROCEDIMENTOS METODOLÓGICOS

O primeiro requisito para desenvolver este trabalho foi compreender os pontos fundamentais de cada etapa envolvida no desenvolvimento de um reconhecedor de fala. Assim foi possível entender a função de cada *software* e os requisitos necessários para desenvolver o trabalho.

A arquitetura típica de um sistema de reconhecimento de fala é ilustrada na Figura 1. De forma simplificada, o sinal da fala é tratado pelo sistema conforme esquematizado no diagrama de blocos. Primeiramente o sinal de entrada é pré-processado e então submetido ao processo de decodificação, onde são utilizados modelos estatísticos de referência, os quais sintetizam o vocabulário do reconhecedor (ANDREÃO, 2001).

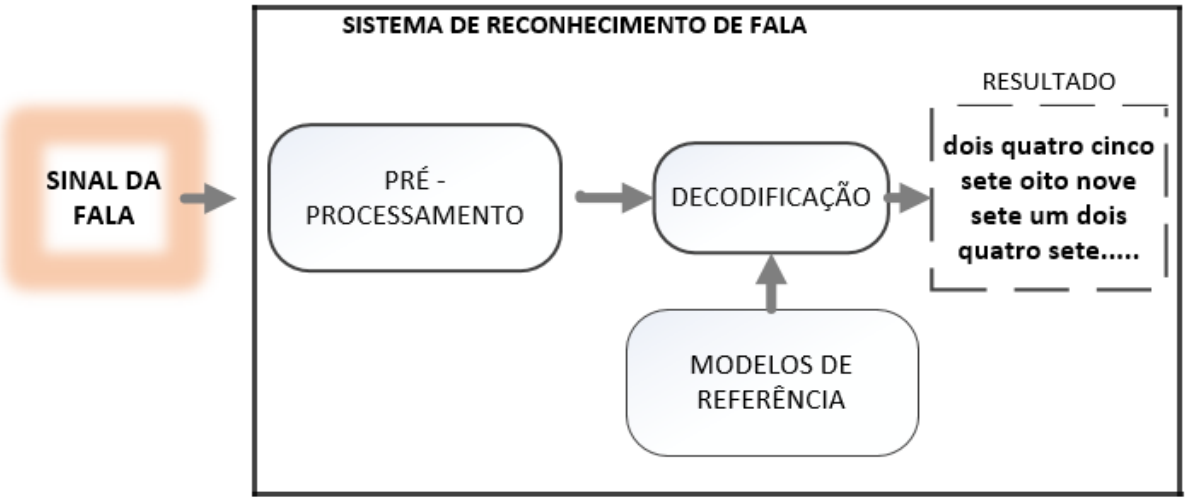


Figura 1 - Arquitetura de Um Sistema de Reconhecimento de Fala.
Fonte: Adaptado de Andreão (2001).

O sistema proposto foi constituído por um vocabulário finito, contendo especificamente os dígitos “0”, “1”, “2”, “3”, “4”, “5”, “6”, “7”, “8” e “9”. O vocabulário foi modelado a partir de um banco de pronúncias de dígitos gravados por diversos locutores e de suas respectivas transcrições.

O HTK consiste em um conjunto de ferramentas utilizadas para construção e treinamento de Modelos Ocultos de Markov (HMM). Seu uso permite o desenvolvimento de modelos acústicos para aplicações em reconhecimento de fala.

O desempenho e as características de um sistema de reconhecimento como, por exemplo, dependência e independência de locutor dependem dos treinamentos realizados. Por essa razão, foi necessário realizar um estudo aprofundado com objetivo de compreender os principais aspectos teóricos e práticos da ferramenta, e assim compreender os pontos fundamentais que envolvem o treinamento dos modelos.

Em conjunto com o sistema de reconhecimento, propôs-se a construção de um protótipo. Para montagem, foi adquirido um modelo do Raspberry Pi, que entre outros recursos, inclui 4 portas USB e pinos de propósito geral (GPIO). A presença do GPIO na placa permite a conexão de circuitos personalizados e, portanto, o uso do Raspberry Pi para o devido acionamento de LEDs simulando a abertura da fechadura eletrônica.

1.6 ESTRUTURA DO TRABALHO

Este trabalho esta estruturado da seguinte forma:

Capítulo 1 – Introdução e apresentação do tema, problemas e premissas, objetivo geral, objetivos específicos, justificativa e procedimentos metodológicos.

Capítulo 2 – Conceitos biológicos da fala, a natureza do som, a produção da fala, descrição do trato vocal e nasal, sistema auditivo e percepção do som, o modelamento fonte-filtro e a representação da escala Mel.

Capítulo 3 – Análise do sinal da fala, detalhamento do processo de extração dos parâmetros, vetores de parâmetros mel-cepstrais.

Capítulo 4 – Reconhecimento de padrões, Modelos Ocultos de Markov, algoritmo de *forward* e *backward,* algoritmo de Viterbi e treinamento dos modelos.

Capítulo 5 – Desenvolvimento do sistema de reconhecimento de fala, preparação dos dados de treinamento, desenvolvimento do modelo acústico, a gramática do reconhecedor.

Capítulo 6 – Implementação do protótipo, o Raspberry Pi, detalhamento e informações do código desenvolvido para controle de usuários.

Capítulo 7 – Considerações finais e sugestões para trabalhos futuros.

2 CONCEITOS BIOLÓGICOS DA FALA

O objetivo de descrever o sistema de produção da fala e o sistema auditivo é representá-los matematicamente de forma a construir um sistema de reconhecimento de fala eficiente. Primeiramente será explicado o funcionamento básico do sistema de produção da fala e o modelo fonte-filtro. Em seguida será explicado o sistema auditivo humano e algumas de suas propriedades, que serão úteis para o modelamento do sistema de reconhecimento de fala.

2.1 A NATUREZA DO SOM

O som é uma sensação produzida no sistema auditivo pela ação de ondas mecânicas que alcançam a membrana timpânica fazendo-a vibrar. No ar, as ondas sonoras são ondas longitudinais de pressão que se propagam por meio de sucessivas compressões e rarefações. Durante a propagação da onda não há transporte de matéria, as moléculas de ar oscilam paralelamente à direção de propagação do sinal, transportando somente energia (HUANG et al., 2001).

2.2 A PRODUÇÃO DA FALA

A fala é o resultado da modificação dos sons produzidos na região da glote e constitui, em sua essência, uma forma natural da comunicação humana. Por meio da fala, o homem transmite seus pensamentos e recebe informações do ambiente ao seu redor (LIMA, 2011).

O aparelho fonador humano, ilustrado na Figura 2, é um conjunto de estruturas e órgãos envolvidos na produção dos sons da fala.

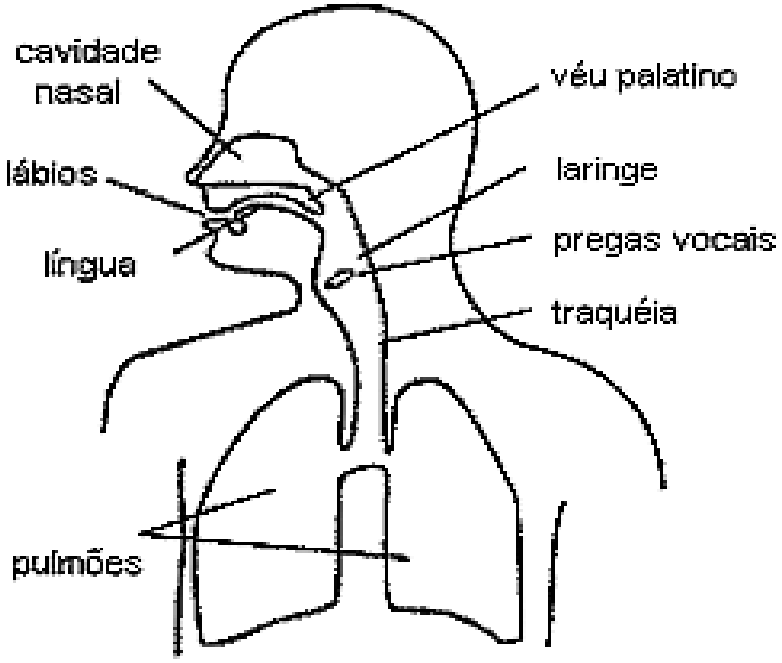


Figura 2 - Principais órgãos do sistema fonador.
Fonte: Simões (1999).

No interior da caixa torácica, os pulmões sofrem expansões e retrações expelindo o ar que passa através da laringe. Na mucosa da laringe formam-se dois pares de pregas: um par é chamado de pregas vestibulares e outro de pregas vocais ou cordas vocais (Figura 3). A abertura entre as pregas vocais denomina-se glote (SIMÕES, 1999).

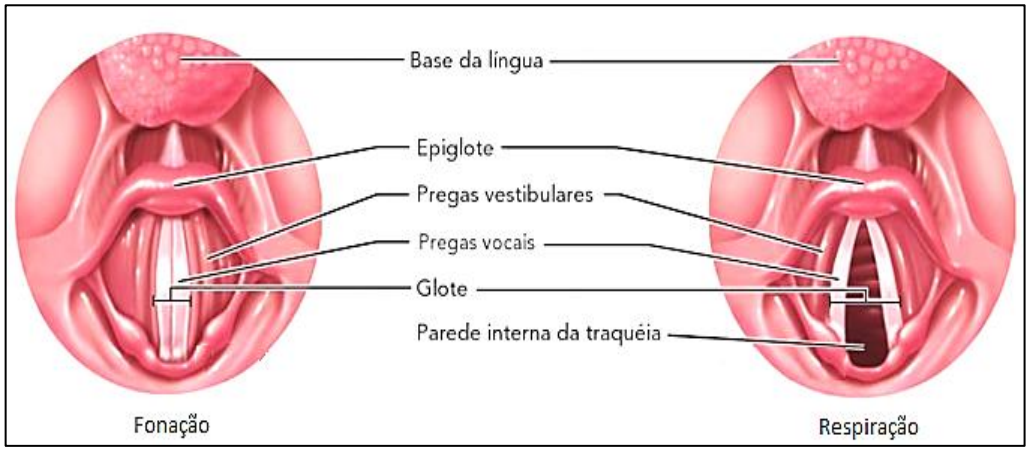


Figura 3 - Representação das pregas vocais.
Fonte: Adaptado de Hoehn e Marieb (2009).

Durante a respiração, as pregas mantêm-se abertas para facilitar o fluxo de ar através da laringe. Na fonação, as pregas se fecham bloqueando a passagem do ar.

O aumento da pressão provocado pelo acúmulo de ar nos pulmões atinge um nível suficiente para forçar a abertura das pregas, permitindo a passagem do ar. Após a pressão nos pulmões ser reduzida, as pregas voltam a se fechar e o ciclo é repetido (HUANG et al., 2001).

A partir desse movimento de abrir e fechar das pregas tem-se o sinal denominado sinal glotal, que consiste de uma série de pulsos e é a principal excitação dos sons sonoros ou vozeados. Naturalmente, as componentes espectrais do sinal glotal decrescem com o aumento das frequências (KENT; READ, 2015).

2.2.1 Trato Vocal e Nasal

O sinal produzido na glote se propaga pela faringe e através da cavidade oral até atingir os lábios, onde é radiado para o espaço. Outra possibilidade é o sinal entrar na cavidade nasal até alcançar as narinas.

O caminho que se estende da glote até os lábios, denominado trato vocal, pode ser modelado como um tubo de ressonâncias de secção transversal não uniforme. As frequências de ressonância de tal tubo são denominadas formantes (PULKKI; KARJALAINEN, 2014). A configuração do trato vocal é determinada por estruturas de articulação como, por exemplo, língua, lábios, véu palatino e mandíbula (SCHAFER; RABINER, 2007).

O trato vocal se conecta à cavidade nasal por meio do véu palatino, o qual atua como uma chave controlando a passagem do ar através da cavidade nasal, para produção de sons nasais (SCHAFER; RABINER, 2007).

2.2.2 Modelamento Fonte Filtro

O modelo, ilustrado na Figura 4, denomina-se fonte-filtro e é tipicamente utilizado para representar a produção da fala. Nele, as componentes espectrais, “U”, do sinal definido na região laríngea vão sendo modificadas à medida que atravessam o

trato vocal. No percurso, as frequências enfatizadas e as frequências atenuadas dependem da configuração dos articuladores, que em conjunto descrevem a função de transferência que o caracteriza (KENT; READ, 2015).

Na mesma figura, “T” representa a função de transferência do trato vocal (mandíbula, lábios e língua) e “R” representa o efeito de radiação, modelado por um filtro em forma de rampa, que tem como objetivo incluir o efeito que ocorre quando o sinal da fala escapa do trato vocal para propagar no espaço. “P” é o espectro de saída resultante, o qual será útil para representar o vetor de parâmetros na implementação do reconhecedor (KENT; READ, 2015).

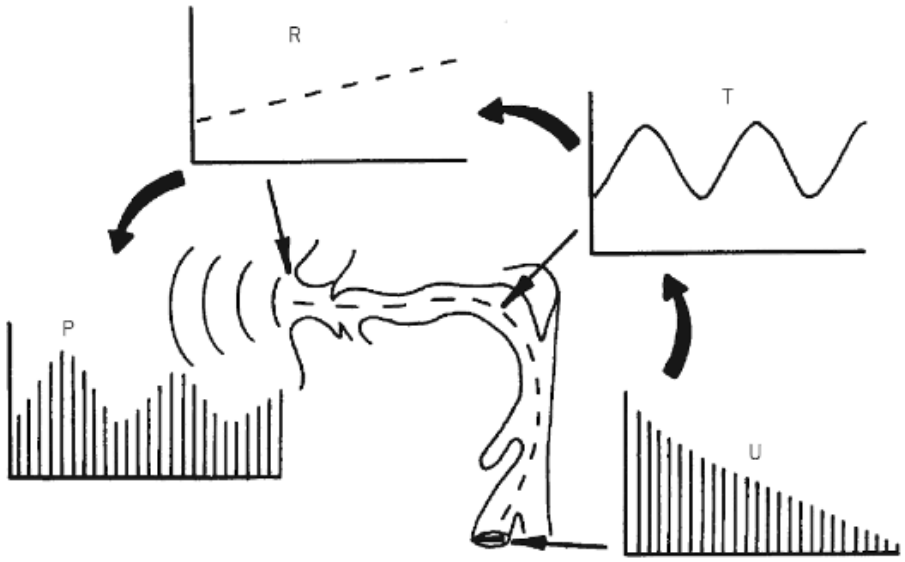


Figura 4 - Diagrama da teoria fonte-filtro para vogais.
Fonte: Adaptado de Kent e Read (2015).

2.3 O SISTEMA AUDITIVO HUMANO

A função do sistema auditivo é converter a energia de uma onda de pressão em estímulos nervosos. Para realizar esta conversão, o sistema conduz energia a partir da membrana timpânica (tímpano) até a cóclea (ouvido interno) onde estimula uma série de células eletricamente sensíveis, chamadas de células ciliadas. Em síntese, as células ciliadas geram impulsos nervosos, em resposta as vibrações da membrana timpânica, que agem em células localizadas no córtex auditivo (OKUNO; CALDAS; CHOW, 1986).

2.3.1 A Percepção do Som

O ouvido humano é dividido anatomicamente em três partes: ouvido interno, ouvido médio e ouvido externo.

O ouvido externo é formado pela aurícula e também por um canal ressonador, aproximadamente cilíndrico, denominado meato acústico ou canal auditivo. A função da aurícula é direcionar as ondas sonoras para o meato acústico externo, o qual culmina na membrana timpânica ou simplesmente tímpano (HOEHN; MARIEB, 2009). Entre a membrana timpânica e a cóclea (ouvido interno), existe um sistema constituído por três ossículos responsáveis pela condução da energia sonora. Os ossículos são o martelo, a bigorna e o estribo, conforme ilustrado na Figura 5 (GUYTON; HALL, 2006).

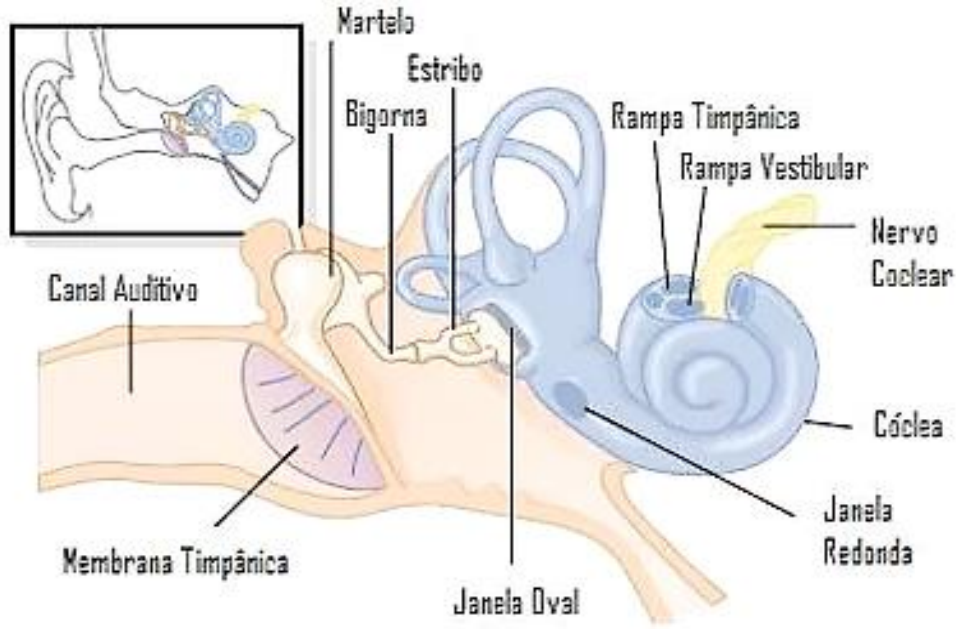


Figura 5 - Representação do sistema auditivo humano.
Fonte: Guyton e Hall (2006).

O martelo, com o cabo fixado na membrana timpânica, comunica-se com a bigorna por minúsculos ligamentos. A extremidade oposta da bigorna articula-se com a base do estribo na janela oval (GUYTON; HALL, 2006).

Em resposta às vibrações da membrana timpânica o martelo e a bigorna oscilam, fazendo com que o estribo se mova para dentro e para fora na abertura da ja-

nela oval, transmitindo a energia sonora da membrana timpânica para o fluído da cóclea (OKUNO; CALDAS; CHOW, 1986).

A cóclea é uma câmara óssea em espiral constituída por três tubos espiralados; a rampa do vestíbulo, rampa média e rampa do tímpano (Figura 5). A rampa média, também denominada tubo coclear, percorre o centro da cóclea. No ducto coclear, sobre a membrana basilar, localiza-se uma estrutura complexa denominada órgão de Corti (HOEHN; MARIEB, 2009).

O órgão de Corti é um transdutor, constituído por um conjunto de células ciliadas eletricamente sensíveis. Tais células geram impulsos nervosos em resposta a estímulos produzidos pela ação de ondas de pressão, que atingem a membrana timpânica fazendo-a vibrar. O órgão de Corti é a estrutura responsável pela percepção consciente dos diferentes sons da fala (HOEHN; MARIEB, 2009).

2.3.2 A Escala Mel

Fisiologicamente, a percepção do som não é linear em todas as faixas de frequências. Até aproximadamente 1000 Hz, há uma percepção linear que se torna logarítmica à medida que a frequência da fonte aumenta. A escala Mel é uma escala perceptual, motivada por este comportamento não linear do sistema auditivo humano (JUANG; RABINER, 1993).

Para cada som real, com frequência ($f_{[Hz]}$) medida em Hertz, utiliza-se a Equação (1) para mapear uma frequência subjetiva ($f_{[mel]}$) na escala Mel. Tal frequência subjetiva corresponde a real percepção do sistema auditivo humano (JUANG; RABINER, 1993).

$$f_{[mel]} = 2595 \log\left(1 + \frac{f_{[Hz]}}{700}\right) \quad (1)$$

Conforme demonstrado na Figura 6, o mapeamento entre as escalas de frequência Hertz e Mel se mantem aproximadamente linear até 1000 Hz e então passa a ser logarítmico.

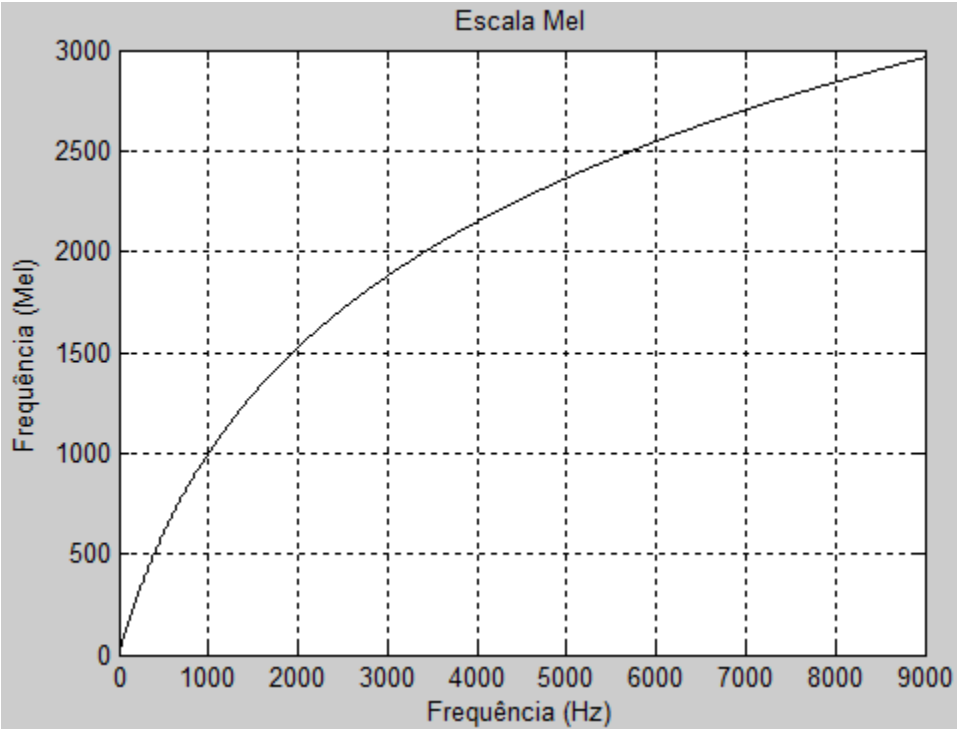


Figura 6 - Mapeamento de frequências em Hz para a escala Mel.
Fonte: Autoria própria.

Justifica-se a escolha da escala Mel devido ao seu uso em trabalhos de reconhecimento de fala. Há outras escalas com diferentes equacionamentos do mesmo tipo, a escolha é baseada em conteúdos perceptuais e, portanto, sugestiva.

2.3.2.1 Representação da escala Mel

Variações exponenciais entre duas frequências de excitação são percebidas pelo sistema auditivo humano como variações lineares. Esta característica perceptual do sistema auditivo humano é utilizada no modelamento de sistemas de reconhecimento de fala.

Para representar a função da cóclea, utiliza-se um banco de filtros que é um conjunto de filtros triangulares passa-faixa dispostos ao longo da escala Mel, conforme representado na Figura 7.

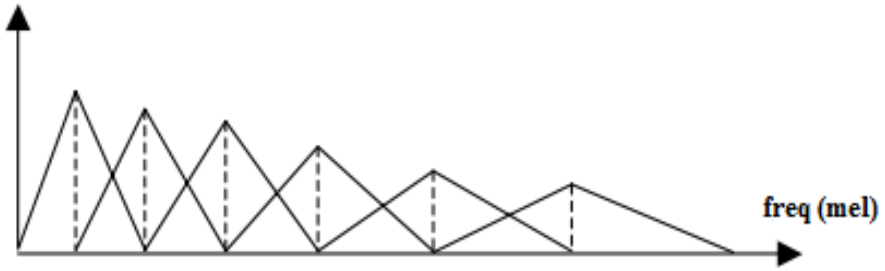


Figura 7 - Filtros triangulares utilizados na parametrização do sinal.
Fonte: Adaptado de Huang et al. (2001).

Observa-se em baixas frequências, que o espaçamento entre os filtros é aproximadamente linear e que a largura de banda de cada filtro aumenta conforme o aumento das frequências. O banco de filtros é utilizado no modelamento de sistemas de reconhecimento de fala para computar a energia média do espectro do sinal em torno da frequência central, na faixa de frequência de cada filtro. Esta aplicação corresponde a um dos passos do processo de extração dos coeficientes mel-cepstrais do sinal da fala, o qual será descrito no capítulo a seguir (HUANG et al., 2001).

3 ANÁLISE DO SINAL DE FALA

O capítulo anterior abordou alguns aspectos teóricos fundamentais que servirão como base para a compreensão de assuntos tratados neste e nos próximos capítulos.

O desenvolvimento de um sistema de reconhecimento de fala divide-se em duas etapas. Uma delas baseia-se em algoritmos probabilísticos para treinamento de modelos e reconhecimento de padrões, a outra é a codificação do sinal. A Figura 8 mostra as duas etapas com destaque para a etapa de codificação, a qual será descrita neste capítulo.

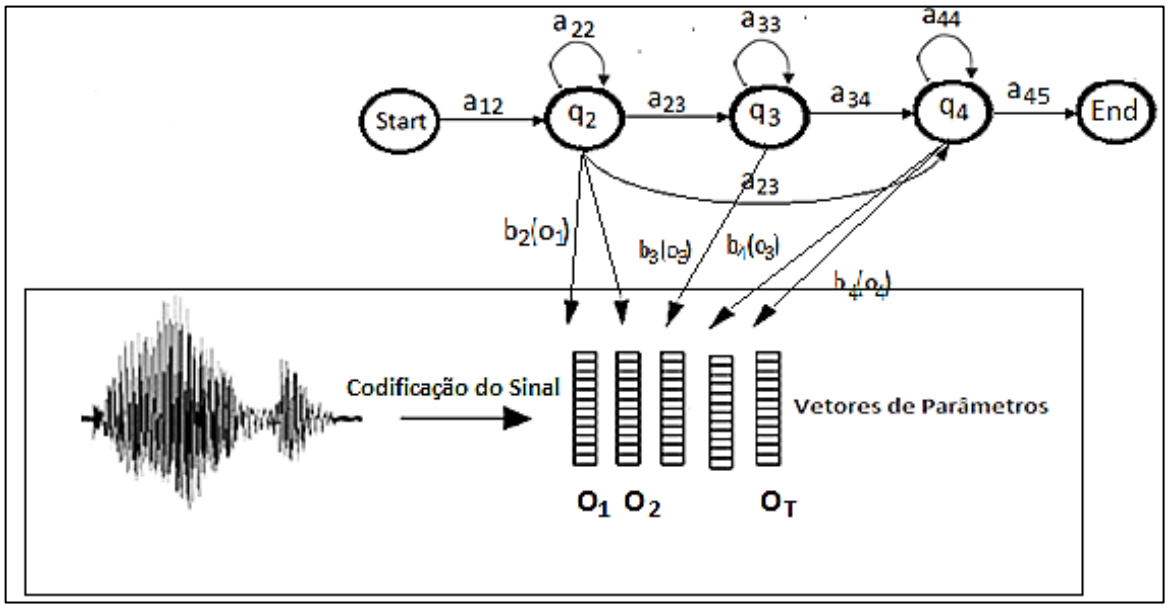


Figura 8 - A codificação do sinal da fala.

Fonte: Adaptado de Hosn (2006).

3.1 CODIFICAÇÃO DO SINAL

Em sistemas de reconhecimento de fala, converte-se o sinal da fala em uma sequência de vetores formados por parâmetros acústicos extraídos do sinal. Este processo, representado na Figura 9, denomina-se codificação do sinal da fala.

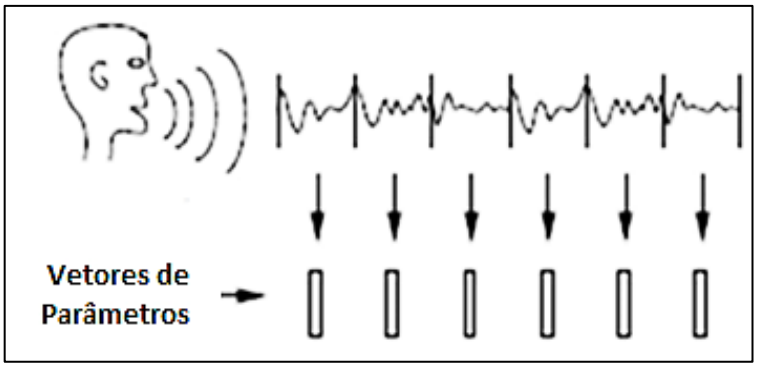


Figura 9 - Esquema da codificação do sinal.
Fonte: Adaptado de Young et al. (2006).

Existem outros métodos de codificação do sinal. A abordagem utilizada neste trabalho é amplamente empregada, baseia-se na aplicação da Transformada Rápida de Fourier (FFT) e da escala Mel para modelar características perceptuais do sistema auditivo humano.

O objetivo é extrair de um curto trecho do sinal da fala um conjunto de parâmetros denominados Coeficientes Mel-Cepstrais (MFCC). Estes parâmetros carregam informações espectrais importantes, que representam bem a informação sonora em um curto segmento do sinal da fala.

Conforme esquematizado na Figura 10, o processo macro de parametrização do sinal da voz explora princípios auditivos e também fisiológicos implícitos no mecanismo de produção da fala (QUATIERI, 2002).

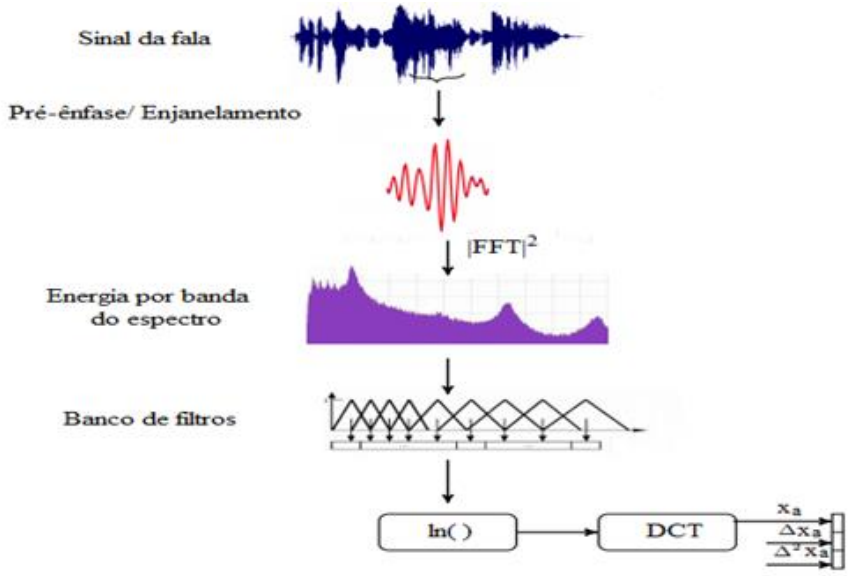


Figura 10 - Esquema da parametrização do sinal da fala.
Fonte: Adaptado de Wang (2015).

Inicialmente o processo divide-se em etapas, ilustradas na Figura 11, que realizam a pré-ênfase, segmentação, enjanelamento e extração dos parâmetros MFCC. No HTK[1], sistema usado neste trabalho, os parâmetros são obtidos com o uso de ferramentas de codificação que realizam cada uma das etapas descritas a seguir.

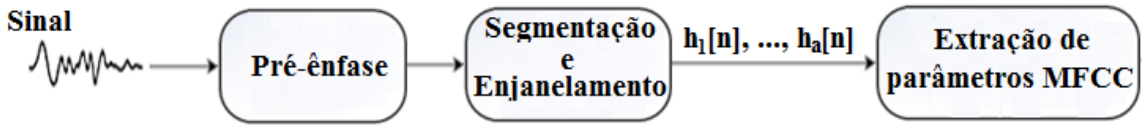


Figura 11 - Etapas envolvidas na codificação do sinal da fala.
Fonte: Adaptado de Dias (2000).

3.1.1 Filtro Pré-Ênfase

Primeiramente, o sinal da fala digitalizado passa através de um sistema de primeira ordem, denominado filtro de pré-ênfase, o qual permite o modelamento apropriado de frequências formantes de diferentes intensidades e tem como objetivo melhorar a eficiência e a precisão do posterior processo de extração de parâmetros mel-cepstrais do sinal (DENG; O'SHAUGHNESSY, 2003).

No domínio do tempo, a saída s'[n] do filtro está relacionada com a entrada s[n], através da Equação (2), onde o coeficiente indicado por "a_{pre}" permite configurar o nível da pré-ênfase no sinal (SIGMUND, 2003).

$$s'[n] = s[n] - a_{pre}s[n-1], \qquad 0{,}95 \leq a_{pre} \leq 0{,}99 \tag{2}$$

Segundo Sigmund (2003), duas explicações são normalmente utilizadas para elucidar o uso da pré-ênfase como parte do processo de parametrização do sinal da voz. Uma delas é que o filtro permite compensar o efeito de decaimento das componentes espectrais em conjunto com o efeito de radiação, que ocorre quando o sinal da fala escapa do trato vocal para radiar no espaço. Em segundo lugar, o sistema

[1] Conjunto de ferramentas utilizadas para construir e manipular Modelos Ocultos de Markov.

auditivo é mais sensitivo na região do espectro acima de 1 kHz, a qual é enfatizada pelo filtro pré-ênfase.

3.1.2 Segmentação e Enjanelamento

A abordagem comumente utilizada no processo de parametrização do sinal da fala baseia-se na análise do sinal em um curto espaço de tempo. Na produção dos diferentes sons da fala, a configuração do trato vocal se altera lentamente ao longo do tempo e tende a ser relativamente constante em períodos entre 10 ms e 20 ms. Por esta razão, o sinal pré-enfatizado é analisado em segmentos suficientemente curtos, onde suas características espectrais permanecem aproximadamente estacionárias (HOLMES; HOLMES, 2001).

A Figura 12 mostra o processo de segmentação, onde o comprimento de cada segmento é determinado pela duração (em milissegundos) da janela aplicada. Na prática, empregam-se janelas em torno de 25 ms (YOUNG et al., 2006).

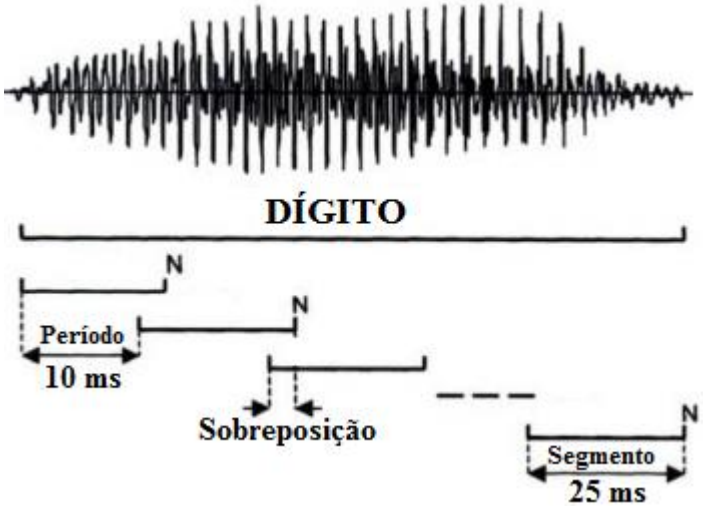


Figura 12 - A conversão do sinal da fala em segmentos sobrepostos.
Fonte: Adaptado de Sigmund. (2003).

Conforme a Figura 12, os segmentos são extraídos do sinal da fala ao aplicar sucessivas janelas espaçadas no tempo pelo período de atualização entre janelas adjacentes. Tal período determina a taxa de segmentos do sinal extraídos a cada

segundo (DENG; O'SHAUGHNESSY, 2003). Um valor tipicamente configurado para o período de atualização é igual a 10 ms (YOUNG et al., 2006).

3.1.2.1 A janela de Hamming

Quando se extrai um segmento do sinal, implicitamente o segmento foi enjanelado por uma janela retangular. Pelo fato de começar e terminar bruscamente, a janela retangular degrada as extremidades dos segmentos, introduzindo efeitos indesejáveis no conteúdo espectral (HOLMES; HOLMES, 2001).

Para atenuar tais degradações utilizam-se janelas suaves como, por exemplo, a janela de Hamming. A Equação (3) representa matematicamente uma janela de Hamming, onde "N" é o número de amostras na janela.

$$W[n] = \begin{cases} 0{,}54 - 0{,}46\cos\left(\frac{2\pi n}{N-1}\right), & 0 \le n \le N-1 \\ 0, & caso\ contrário \end{cases} \quad (2)$$

A Figura 13 mostra uma janela de Hamming, com "N" igual a 251.

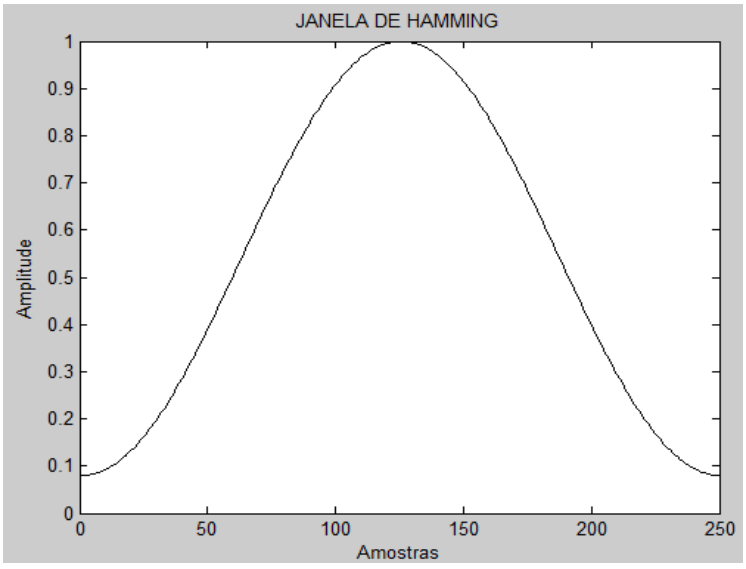


Figura 13 - Representação da janela de Hamming.

Fonte: Autoria própria.

3.1.3 Extração de Parâmetros MFCC

A etapa posterior ao enjanelamento do sinal denomina-se extração dos parâmetros mel-cepstrais. Nesta etapa, cada janela ($h_1[n]$, $h_2[n]$,..., $h_a[n]$,...), resultante do processo de enjanelamento, gera um vetor denominado vetor de parâmetros, o qual será utilizado posteriormente para treinamento do sistema e reconhecimento da fala.

A extração dos parâmetros mel-cepstrais compreende a sequência de passos esquematizados na Figura 14.

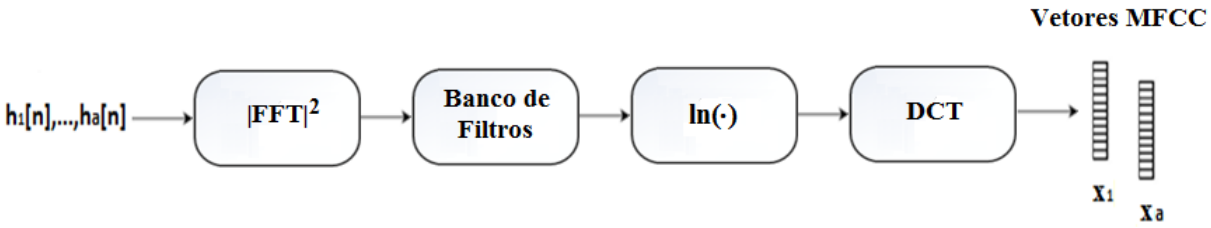


Figura 14 - Processo da extração de parâmetros MFCC.
Fonte: Adaptado de Dias (2000).

3.1.3.1 Cálculo da energia por banda do espectro

Conforme a Figura 14, primeiramente calcula-se a energia por banda do espectro de cada janela do sinal, computando o quadrado do módulo da FFT. A Equação (4) mostra esta aplicação, onde $h_a[n]$ é a a-ésima janela e “N” o respectivo número de amostras.

$$|H_a[k]|^2 = \left| \sum_{n=0}^{N-1} h_a[n] e^{-\frac{j2\pi nk}{N}} \right|^2, \qquad 0 \leq k < N \tag{4}$$

3.1.3.2 Aplicação da escala Mel

Neste passo, define-se um banco de filtros triangulares, $X_m[k]$, dispostos ao longo da escala Mel, onde m = 1, 2,..., ”M” refere-se ao índice do m-ésimo filtro no banco. A função do banco de filtros é modelar a função da cóclea (HUANG et al., 2001).

Cada filtro é aplicado sobre o espectro de energia do sinal e o resultado, uma sequência de coeficientes, é somada para calcular a energia do sinal na faixa de frequência de cada filtro.

Assim, através de “M” filtros ter-se-á “M” coeficientes de energia. Na literatura fala-se de “M” entre 24 e 40 (HUANG et al., 2001). A equação (5) mostra esta aplicação, onde “N” é o tamanho da FFT.

$$S_a[m] = \sum_{k=0}^{N-1} |H_a[k]|^2 \, Xm[k] \,, \qquad 0 \leq m \leq M \tag{5}$$

3.1.3.3 Logaritmo da energia por banda do espectro

Neste passo, o resultado obtido após aplicação da Equação (5) é utilizado para calcular, com o uso da Equação (6), o logaritmo da energia de saída para o m-ésimo filtro do banco.

$$V_a[m] = \ln(S_a[m]) \,, \qquad 1 \leq m \leq M \tag{6}$$

3.1.3.4 Parâmetros estáticos do sinal do fala

Ainda segundo o diagrama de blocos esquematizado na Figura 14, após o cálculo do logaritmo da energia ser executado, aplica-se a DCT (Transformada Discreta do Cosseno) para extração dos chamados parâmetros mel-cepstrais.

Os parâmetros são calculados utilizando a Equação (7), onde "M" é o número de filtros e $V_a[m]$ é o logaritmo da energia acumulada no m-ésimo filtro para a a-ésima janela do sinal da fala.

$$c_a[n] = \sum_{m=0}^{M-1} V_a[m] \cos\left(\frac{\pi n\left(m+\frac{1}{2}\right)}{M}\right), \qquad 0 \leq n < M\ ,\ 1 \leq m < M \tag{7}$$

De forma similar ao cálculo dos coeficientes de energia, aplicando a Equação (7) para "M" filtros triangulares, obtém-se "M" parâmetros mel-cepstrais.

Por conta dos efeitos da DCT, de concentração de energia nos primeiros coeficientes, utilizam-se um número reduzido de componentes, "C_{mfcc}", para compor o vetor. Geralmente o valor de "C_{mfcc}" é 13 (HUANG et al., 2001).

O elemento $c_a[0]$, calculado em (7), é normalmente descartado ou substituído por uma medida de menor variabilidade (KNILL; YOUNG, 1997). Usualmente substitui-se este elemento pelo logaritmo de energia do sinal, calculado no domínio do tempo ao longo das amostras da janela. A Equação (8) mostra esta aplicação onde $h_a[n]$ representa a a-ésima janela do sinal analisado.

$$E = log \sum_{n=0}^{N-1} (h_a[n])^2 \tag{8}$$

Após os cálculos serem concluídos, são extraídos de cada janela 12 parâmetros mel-cepstrais originais e uma componente de energia “E”, resultando em 13 parâmetros estáticos do sinal da fala.

3.1.3.5 Parâmetros dinâmicos do sinal do fala

É possível aprimorar o desempenho do sistema de reconhecimento de fala adicionando ao conjunto de parâmetros estáticos outros parâmetros, denominados parâmetros dinâmicos.

Para cada parâmetro estático (Equação 7), inclusive a componente de energia (Equação 8), calcula-se um parâmetro delta (velocidade) e um parâmetro de delta-delta (aceleração) (JURAFKSY; MARTIN, 2008).

Os parâmetros delta e aceleração são anexados aos parâmetros estáticos em uma tentativa de melhorar a suposição da independência entre os vetores de parâmetros, associada aos Modelos Ocultos de Markov (GALES; YOUNG, 2007).

Os parâmetros delta, Δc_a, são calculados com o uso da Equação (9), em termo dos vetores de parâmetros estáticos, c_a.

$$\Delta c_a = \frac{\sum_{\delta=1}^{2} \delta (c_{a+\delta} - c_{a-\delta})}{2 \sum_{\delta=1}^{2} \delta^2} \tag{9}$$

Por sua vez, os parâmetros de aceleração, $\Delta^2 c_a$, são calculados com uso da Equação (10), em termo dos parâmetros delta, Δc_a.

$$\Delta^2 c_a = \frac{\sum_{\delta=1}^{2} \delta (\Delta c_{a+\delta} - \Delta c_{a-\delta})}{2 \sum_{\delta=1}^{2} \delta^2} \tag{10}$$

Após os cálculos terem sido concluídos, cada janela gera 13 parâmetros estáticos originais, 13 parâmetros delta e 13 parâmetros de aceleração, resultando em

39 parâmetros do sinal da fala. Estes parâmetros podem ser agregados para formar um vetor “x_a” denominado vetor de parâmetros, conforme representado em (11) (JURAFKSY; MARTIN, 2008).

$$x_a = [c_a \quad \Delta c_a \quad \Delta c_a^2] \tag{11}$$

O vetor “x_a” carrega parâmetros espectrais estáticos, dinâmicos e de energia da a-ésima janela do sinal. Os parâmetros representam informações do sinal acústico que são caracterizadas com maior precisão no domínio da frequência.

4 RECONHECIMENTO DE PADRÕES

O capítulo anterior abordou a análise do sinal, ou seja, o processo de codificação da informação sonora em uma sequência de vetores de parâmetros. Conforme ilustrado na Figura 15, os vetores de parâmetros constituem um processo observável para cada entrada de voz, onde o objetivo do decodificador é realizar o mapeamento da sentença de palavras subjacente à sequência de características acústicas observadas (KNILL; YOUNG, 1997).

O modelo probabilístico, na parte superior da Figura 15, denomina-se Modelo Oculto de Markov (HMM) e tem como propósito executar a decodificação do sinal. HMMs são frequentemente utilizados no modelamento de reconhecedores de fala porque produzem bons resultados como manipuladores de aspectos estatísticos e sequências da informação sonora codificada (OLIVEIRA, 2001).

Aqui e no restante deste capítulo, as observações "o_t" correspondem aos vetores de parâmetro "x_a" do capítulo anterior. Esta mudança no texto se deve ao fato de que a literatura de reconhecimento de padrões usa "o_t" para generalizar o vetor de análise.

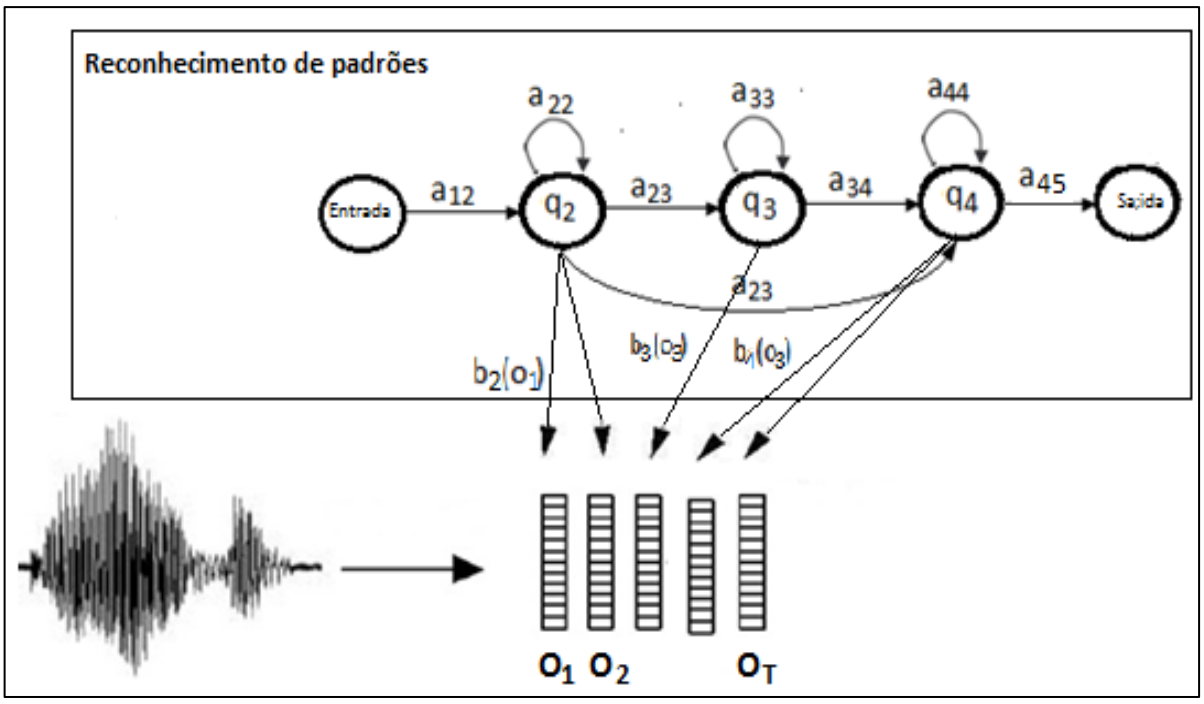


Figura 15 - Reconhecimento de padrões.

Fonte: Adaptado de Hosn (2006).

4.1 MODELOS OCULTOS DE MARKOV

Um Modelo Oculto de Markov (HMM) é uma ferramenta usada para modelar estatisticamente fenômenos aleatórios observáveis no tempo (GHAHRAMANI, 2001). Tais modelos sintetizam adequadamente a evolução temporal do sinal da voz a representando por meio de um processo paramétrico duplamente estocástico (LI et al., 2015).

Em cada instante de tempo “t”, há um processo não observável, formado por uma sequência de estados, que é fonte geradora de um segundo processo, o qual pode ser observado por meio de uma sequência de emissões O = {o_1, o_2,..., o_T} que são reproduzidas conforme a evolução dos estados do modelo (WANG, 2015).

No instante de tempo corrente, ambos os processos citados encapsulam a informação necessária para predizer o futuro do processo, independentemente de estados e observações anteriores, satisfazendo assim, uma propriedade de Markov (GHAHRAMANI, 2001).

A evolução nos estados de um HMM não pode ser diretamente observada, permanecendo subjacente à sequência de observações O = {o_1, o_2,..., o_T}. Por esta razão, quando se “ouve” um fone, não se sabe em qual estado o modelo se encontra, dando origem ao termo “oculto” (SILVA, 1999).

4.1.1 Parametrização de Um Modelo Oculto de Markov

Um HMM é parametrizado da seguinte forma:

- Um conjunto de “N” estados ocultos:

$$Q = \{q_1, q_2, \dots, q_N\} \tag{12}$$

- Um conjunto de probabilidades de transição, a_{ij}, armazenadas em uma matriz A = {a_{ij}}:

$$a_{ij} = P(q_t = j \backslash q_{t-1} = i), \quad 1 \leq i, j \leq N, \quad 0 \leq a_{ij} \leq 1 \tag{13}$$

Onde a_{ij} representa a probabilidade de transição do estado “i” para o estado “j”. Para processos de Markov de primeira ordem assume-se que a probabilidade de transição para um determinado estado “j”, depende somente do estado atual “i” (OLIVEIRA, 2001).

- Uma distribuição de probabilidade inicial:

$$\pi_i = P(q_1 = i), 1 \leq i \leq N \tag{14}$$

Onde π_i é a probabilidade da sequência de transições iniciar no estado “i”.

- Uma distribuição de probabilidade de observação associada a cada estado emissor do modelo:

$$b_j(o_t) = P(o_t \backslash q_t = j) \qquad 1 \leq i, j \leq N, \qquad 1 \leq i, j \leq N \tag{15}$$

Onde $b_j(o_t)$ é a probabilidade do HMM emitir a observação “o_t” dado o estado “j” no instante “t”.

4.1.2 Representação de Modelos Ocultos de Markov

A notação (16) é geralmente utilizada para representar um HMM, onde “λ” é o modelo e A, B e π são os seus parâmetros.

.

$$\lambda = (A, B, \pi) \tag{16}$$

O sinal da fala é uma sequência no tempo, a Figura 16 mostra um HMM com 5 estados frequentemente utilizado no modelamento do sinal da fala (WANG, 2015). A topologia representada denomina-se *left-right* porque as restrições estocásticas de

transição, a_{ij}, permitem apenas a evolução entre os estados da esquerda para a direita, conforme indicado pelas setas direcionais na mesma figura. Ainda observa-se na Figura 16, a existência de três estados emissores (q_2, q_3 e q_4). Também é observada a existência de dois estados não emissores (q_1 e q_5), reservados para entrada e saída do modelo. Os estados não emissores são projetados para facilitar a conexão entre diferentes HMMs. Isto permite, por exemplo, que HMMs baseados em fones possam ser conectados para o modelamento de diferentes pronúncias (YOUNG, 1996).

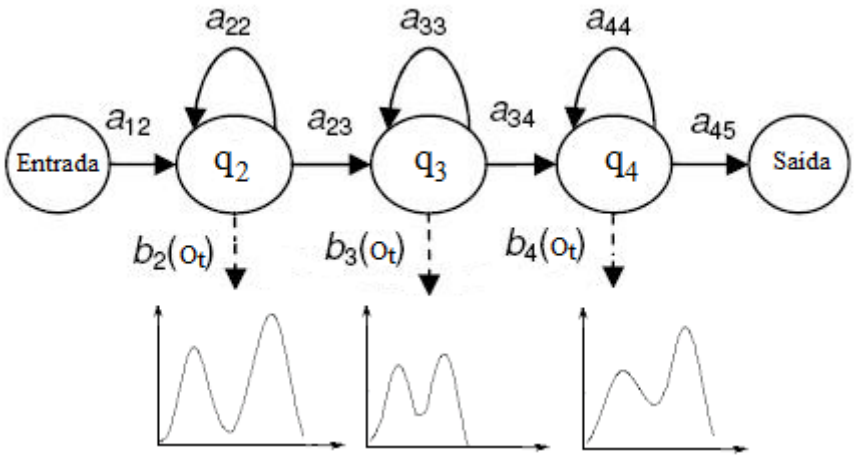


Figura 16 - Modelo HMM com três estados emissores.
Fonte: Adaptado de Gales e Young (2007).

4.1.3 Distribuição de Probabilidades de Observação

Para realizar os cálculos de observação, associa-se a cada estado emissor do HMM uma distribuição de probabilidades, denotada por $b_j(o_t)$. A distribuição probabilística é modelada de acordo com a natureza estocástica, continua ou discreta, da fonte geradora. Conforme a Equação (17), uma importante classe de distribuição utilizada em sistemas de reconhecimento da fala, baseia-se na soma ponderada de "M" Gaussianas multivariadas, ou seja, modelo de misturas Gaussianas (JURAFKSY; MARTIN, 2008).

O uso de “M” componentes Gaussianas, permite descrever com maior qualidade dados físicos que exibem multimodalidade, e que são pouco adequados para uma componente Gaussiana simples (DENG; YU, 2015).

Na Equação (17) “o_t” representa o vetor de parâmetros sendo modelado e C_{jm} refere-se ao coeficiente de ponderação associado a m-ésima componente Gaussiana. Os parâmetros μ_{jm} e Σ_{jm} são respectivamente um vetor de médias e uma matriz de covariância, associados ao estado emissor “j” e a m-ésima componente Gaussiana da mistura (RABINER, 1989).

$$b_j(o_t) = \sum_{m=1}^{M} C_{jm} \frac{1}{\sqrt{(2\pi)^n \left|\Sigma_{mj}\right|}} \mathrm{e}^{[(o_t-\mu_{mj})^T \Sigma_{mj}{}^{-1}(o_t-\mu_{mj})]} \tag{17}$$

O vetor μ_{jm} associa uma média a cada dimensão do vetor de parâmetros observado. A matriz de covariância Σ_{jm} captura a variância de todas as dimensões do vetor de parâmetros em sua diagonal principal, bem como a covariância entre duas dimensões nos outros elementos da matriz (JURAFKSY; MARTIN, 2008).

4.2 TREINAMENTO DE MODELOS OCULTOS DE MARKOV

A abordagem estatística utilizada na implementação de um sistema de reconhecimento de fala envolve o uso de dois conhecimentos estocásticos que podem ser modelados separadamente, o modelo acústico e o modelo de linguagem.

O modelo de linguagem independe da observação do sinal de entrada e tem como objetivo captar restrições sintáticas e pragmáticas da linguagem em consideração (NEY, 1997). Já o modelo acústico, proporciona um eficiente método para calcular probabilidades de observação, dado uma sentença hipotética de palavras (YOUNG, 1996).

Em reconhecimento da fala, o processo de avaliação é realizado de forma a garantir uma ótima interação entre as duas fontes citadas de conhecimento. O método de aprendizagem deriva em um problema matemático de otimização, cujos detalhes dependem da topologia dos modelos e do critério de treinamento selecionado para a implementação do sistema (NEY, 1997).

No processo de estimação dos HMMs, as observações adotadas como base denominam-se exemplos ou sequências de treinamento e são geradas a partir de um banco de pronúncias gravado por diversos locutores (JUANG; RABINER, 1991).

Em principio, estimação do modelo acústico baseia-se no ótimo alinhamento entre HMMs e exemplos de treinamentos. Neste processo, utiliza-se um dicionário fonético para guiar um eficiente método de treinamento denominado algoritmo Baum-Welch, o qual produz bons resultados no modelamento de sistemas de reconhecimento de fala e será melhor descrito no item a seguir (SCHAFER; RABINER, 2007)

4.2.1 O Algoritmo de Baum-Welch

O algoritmo de Baum-Welch é um procedimento iterativo de re-estimação, empregado na etapa de treinamento dos modelos. Dado um conjunto de características acústicas, o treinamento busca ajustar os parâmetros de transição entre estados (a_{ij}) e de observação (μ_{jm}, Σ_{jm}, C_{jm}). Sinteticamente, objetivo é maximizar a probabilidade de um dado autômato “λ” reproduzir uma dada sequência de vetores $O = \{o_1, o_2,..., o_T\}$ (RABINER, 1989).

O algoritmo faz uso das variáveis de *forward* e *backward* e baseia-se em métodos estatísticos para encontrar a máxima verossimilhança existente entre os parâmetros dos modelos (SILVA, 1999). A etapa seguinte, chamada de re-estimação, consiste na maximização das probabilidades de transição e de observação, a partir das variáveis de *forward* e *backward*, de forma a estimar parâmetros mais precisos para os HMMs.

Em cada iteração, os modelos “λ” são testados e automaticamente atualizados para um novo conjunto de modelos. No algoritmo de Baum-Welch, este procedimento é repetido até um determinado critério de término ser atendido (JUANG; RABINER, 1991).

4.2.1.1 A variável de *forward*

A variável de *forward*, definida na Equação (18), é à probabilidade de emissão da sequência de observações parciais O = {o_1, o_2,..., o_t}, terminando no estado “i” e no instante “t”, dado o modelo gerador “λ” (BILMES, 1998).

$$\alpha_t(i) = P(o_1 o_2 \dots o_t, q_t = i \backslash \lambda) \tag{18}$$

No algoritmo de *forward*, a soma das probabilidades é computada considerando todas as possíveis combinações de estados, que possam gerar as “t” primeiras observações da sequência e que terminam no estado “i” do modelo (JURAFSKY; MARTIN, 2008).

Para a topologia de HMM ilustrada na Figura 15, a variável de *forward* pode ser calculada recursivamente utilizando a Equação (19).

$$\alpha_t(j) = \left[\sum_{i=2}^{N-1} \alpha_{t-1}(i)\, \mathrm{a}_{ij}\right] b_j(o_t); \quad 1 < j < N, \qquad 1 < t \leq T \tag{19}$$

A equação (19) compreende os seguintes passos:

- Inicialização:

$$\alpha_1(1) = 1,$$

$$\alpha_1(j) = \mathrm{a}_{1j} b_j(o_1), \;\; 1 < j < N; \tag{20}$$

Neste passo, os estados são inicializados no instante “t” igual a “1”, com probabilidades iniciais de transição e observação (RABINER, 1989).

- Terminação:

$$P(O\backslash\lambda) = \alpha_T(\mathrm{N}) = \sum_{i=2}^{N-1} \alpha_T(i)\mathrm{a}_{iN} \tag{21}$$

Neste passo, a variável $\alpha_T(N)$ computa a probabilidade do modelo “λ” emitir a sequência de observações O = {o_1, o_2,..., o_T}, considerando todas as combinações de estados permitidas. A partir do estado não emissor “1” até o último estado não emissor do modelo. Conforme indicado na Equação (21), $\alpha_T(N)$ representa a probabilidade de emissão total, dado o modelo gerador “λ” (JURAFSKY; MARTIN, 2008).

4.2.1.2 A variável de *backward*

De forma similar a variável de *forward*, a variável de *backward* (Equação 22) representa a probabilidade de emissão da sequência de observações parcial, posteriores ao instante “t”, dado o estado, “i”, no instante atual (BILMES, 1998). No cálculo da variável de *backward*, considera-se a probabilidade sobre todas as possíveis sequências de estados, que possam gerar as observações O = {o_{t+1}, o_{t+2},..., o_T}, a partir do instante “t+1”, até o final do modelo (JURAFSKY; MARTIN, 2008).

$$\beta_t(i) = P(o_{t+1}o_{t+2}, \ldots o_T\backslash q_t = i, \lambda) \tag{22}$$

De forma similar à variável de *forward* (18), a variável de *backward* (22) pode ser calculada recursivamente utilizando a expressão (23).

$$\beta_t(i) = \sum_{j=2}^{N-1} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \qquad 1 \leq i \leq N-1, \qquad 1 \leq t \leq T; \tag{23}$$

A equação (23) compreende os seguintes passos:

- Inicialização:

$$\beta_T(i) = a_{iN}, \qquad 1 < i < N; \tag{24}$$

Dado uma sequência de observações O = {o_{t+1}, o_{t+2},..., o_T}, os estados são inicializados no instante "T" com a probabilidade de transição para o estado não emissor de saída.

- Terminação:

$$P(O\backslash\lambda) = \beta_1(1) = \sum_{j=2}^{N-1} a_{1j}\, b_j(o_1) \beta_1(i) \tag{25}$$

De forma análoga ao algoritmo de *forward*, a variável $\beta_1(1)$ computa a probabilidade do modelo "λ" gerar a sequência completa de observações {o_1, o_2,..., o_T}. No cálculo consideram-se a totalidade das sequências de estados possíveis, a partir da entrada não emissora "1" até a saída não emissora do modelo. Indica-se esta probabilidade por P(O\ λ), conforme representado na Equação (25) (JURAFSKY; MARTIN, 2008).

4.2.1.3 Equações de re-estimação

As variáveis de *forward* (18) e *backward* (22) são combinadas nas equações de re-estimação de Baum-Welch. Estas equações são empregadas na fase de treinamento de HMMs para estimar parâmetros de transição (a_{ij}) e de emissão (μ_{jm}, Σ_{jm}, C_{jm}).

Os parâmetros podem ser avaliados a partir de um conjunto constituído por “R” dados de treinamento ($O^{(1)}$, $O^{(2)}$, ..., $O^{(R)}$), onde $O^{(r)} = \{o_1^{(r)}, o_2^{(r)},..., o_{Tr}^{(r)}\}$ representa uma sequência de observações singular e “r”, com 1 ≤ r ≤ R, refere-se ao índice da sequência no banco de dados do reconhecedor (LI et al., 2000).

As Equações (26), (27) e (28) são utilizadas para re-estimar parâmetros de transição entre estados, onde o expoente entre parênteses “q” refere-se ao índice do q-ésimo HMM na sequência de conexões (YOUNG et al.; 2006).

- $\hat{a}_{ij}^{(q)}$, transições entre estados emissores do modelo:

$$\hat{a}_{ij}^{(q)} = \frac{\sum_{r=1}^{R} \frac{1}{\alpha^r{}_T(\mathrm{N})} \sum_{t=1}^{T_r-1} \alpha_t^{(q)r}(\mathrm{i}) a_{ij}^{(q)} b_j^{(q)}(o_{t+1}^r) \beta_{t+1}^{(q)r}(j)}{\sum_{r=1}^{R} \frac{1}{\alpha^r{}_T(\mathrm{N})} \sum_{t=1}^{T_r-1} a_t^{(q)r}(i) \beta_t^{(q)r}(i)} \tag{26}$$

onde $1 < i < N$ e $1 < j < N$

- $\hat{a}_{1j}^{(q)}$, transição a partir da entrada não emissora “1”:

$$\hat{a}_{1j}^{(q)} = \frac{\sum_{r=1}^{R} \frac{1}{\alpha^r{}_T(\mathrm{N})} \sum_{t=1}^{T_r-1} \alpha_t^{(q)r}(1) a_{1j}^{(q)} b_j^{(q)}(o_t^r) \beta_t^{(q)r}(j)}{\sum_{r=1}^{R} \frac{1}{\alpha^r{}_T(\mathrm{N})} \sum_{t=1}^{T_r-1} a_t^{(q)r}(i) \beta_t^{(q)r}(i) + a_t^{(q)r}(1) a_{1N_q}^{(q)} \beta_t^{(q+1)r}(1)} \tag{27}$$

onde $1 < j < N$.

- $\hat{a}_{iN}^{(q)}$, transição para a saída não emissora “N”:

$$\hat{a}_{iN_q}^{(q)} = \frac{\sum_{r=1}^{R} \frac{1}{\alpha^r{}_T(\mathrm{N})} \sum_{t=1}^{T_r-1} \alpha_t^{(q)r}(\mathrm{i}) a_{iN_q}^{(q)} b_j^{(q)} \beta_t^{(q)r}(N_q)}{\sum_{r=1}^{R} \frac{1}{\alpha^r{}_T(\mathrm{N})} \sum_{t=1}^{T_r} a_t^{(q)r}(i) \beta_t^{(q)r}(i)} \tag{28}$$

onde $1 < i < N$.

A perspectiva de emissão dos vetores pode ser aperfeiçoada utilizando distribuições baseadas em misturas Gaussianas para modelar probabilidades de observação. Por razão dos estados do HMM estarem escondidos do observador, os processos de *forward* e *backward* consentem que cada vetor seja coligado com todos os estados do HMM por meio da expectativa do modelo, "λ", encontrar-se em algum determinado estado "j" no momento em que o vetor foi apontado (GALES; YOUNG, 2007).

Para estimar parâmetros de observação define-se a variável $L_{tm}(j)$, conforme a Equação 29, onde o expoente "q" é o índice do HMM e "r" refere-se ao índice da sequência de observações (YOUNG et al.; 2006).

$$L_{tm}^{(q)r}(j) = \frac{1}{\alpha_T(\mathrm{N})} U_t^{(q)r}(j) C_{jm}^{(q)} b_{jm}^{(q)}(o_t^r) \beta_t^{(q)r}(j) b_j^*(o_t^r) \tag{29}$$

Onde,

$$U_t^{(q)r}(j) = \begin{cases} a_t^{(q)r}(1)\mathrm{a}_{1j}^{(q)}, & caso\ t = 1 \\ a_t^{(q)r}(1)\mathrm{a}_{1j}^{(q)} + \sum_{i=2}^{N-1} a_{t-1}^{(q)r}(i)\mathrm{a}_{ij}^{(q)}, & caso\ contrário \end{cases}$$

e

$$b_j^*(o_t) = \prod_k b_{jk}(o_{kt})$$

Em síntese, a quantidade $L_{tm}(j)$ computa a probabilidade do modelo "λ" estar no estado "j" e no instante "t", com a m-ésima componente da mistura, após ter observado a sequência de vetores $O^{(r)} = \{o_1^{(r)}, o_2^{(r)}, ..., o_{Tr}^{(r)}\}$. A variável $L_{tm}(j)$ é utilizada nas Equações (30), (31) e (32) para re-estimar respectivamente a média, a variância e o coeficiente de ponderação para cada estado emissor "j" do HMM (YOUNG et al, 2006).

- μ_{jm}, vetor de médias associado a m-ésima componente Gaussiana da mistura no estado "j":

$$\hat{\mu}_{jm} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T_r} L_{tm}^{(q)r}(j)\, o_t^r}{\sum_{r=1}^{R}\sum_{t=1}^{T_r} L_{tm}^{(q)r}(j)} \tag{30}$$

- Σ_{jm}, matriz de covariâncias para a m-ésima componente da mistura no estado "j":

$$\hat{\Sigma}_{jm} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T_r} L_{tm}^{(q)r}(j)(o_t^r - \hat{\mu}_{jm})(o_t^r - \hat{\mu}_{jm})'}{\sum_{t=1}^{T_r} L_{tm}^{(q)r}(j)} \tag{31}$$

onde $(o_t - \hat{\mu}_{jm})'$ é o vetor transposto de $(o_t - \hat{\mu}_{jm})$.

- C_{jm}, coeficiente de ponderação associado m-ésima componente da mistura Gaussiana no estado "j":
-

$$C_{jm} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T_r} L_{tm}^{(q)r}(j)}{\sum_{r=1}^{R}\sum_{t=1}^{T_r} L_t^{(q)r}(j)} \tag{32}$$

4.3 O ALGORITMO DE VITERBI

O algoritmo de Viterbi é um eficiente procedimento de programação dinâmica, usado na etapa de decodificação do sinal da fala. Para cada entrada de voz, o algoritmo age em um HMM em busca da sequência de estados ótima, subjacente a alguma sequência de vetores analisada (JURAFKSY; MARTIN, 2008).

No algoritmo, a resolução de um problema intricado é subdivido e as discrepâncias, existentes entre as observações, são calculadas mediante a aplicação de múltiplas regras de decisões, tomadas ao longo das transições através dos estados (DENG; YU, 2015).

Em síntese, o algoritmo monitora o progresso de um HMM, calculando a solução ótima a partir de probabilidades parciais máximas previamente calculadas e

memorizadas. Assim, o algoritmo computa a pontuação da melhor sequência de estados, para cada entrada acústica recebida.

A fim de encontrar a melhor sequência de estados Q = {q_1, q_2,... q_T}, para uma dada sequência de observações O = {o_1, o_2,..., o_T}, define-se a seguinte quantidade (RABINER, 1989):

$$\delta_t(j) = \max_{q_1 q_2 \ldots q_{t-1}} P(q_1 q_2 \ldots q_{t-1},\ q_t = j,\ o_1 o_2 \ldots o_t \backslash \lambda) \tag{33}$$

A quantidade $\delta_t(j)$ determina a probabilidade do modelo "λ" estar no estado "j" e no instante "t", depois de passar através da melhor sequência de estados anteriores {q_1, q_2,... q_{t-1}}. A variável $\delta_t(j)$ evolui calculando probabilidades parciais máximas e monitorando, em cada instante de tempo, os estados que produziram tais probabilidades (RABINER, 1989). Neste contexto, cada estado do HMM é um precursor hipotético do melhor caminho possível (DENG; YU, 2015).

No passo do algoritmo denominado *path backtracking*, a sequência de estados ocultos é recuperada, rastreando-a a partir do último estado oculto do modelo. O algoritmo de Viterbi compreende o conjunto de passos descritos a seguir:

- Inicialização:

$$\delta_1(i) = \pi_i b_i(o_1), \qquad 1 < i < N \tag{34}$$
$$\psi_1(i) = 0;$$

Neste passo, calcula-se probabilidades parciais para todos os estados do HMM no instante "t" igual a 1. A variável ψ é uma variável auxiliar, utilizada para recuperar, posteriormente, a sequência de estados ocultos (RABINER, 1989).

- Recursão:

$$\delta(i) = \max_{1<i<N} \left[\delta(i)_{t-1} a_{ij}\right] b_j(o_t), \qquad 2 \le t \le T, \qquad 1 < j < N \tag{35}$$

$$\psi_t(j) = \arg\max_{1<i<N} \left[\delta_{t-1}(i) a_{ij}\right], \qquad 2 \le t \le T, \qquad 1 < j < N \tag{36}$$

Neste passo, são calculadas probabilidades ótimas parciais a partir do instante “t” igual a 2. Para recuperar a sequência de estados ocultos é necessário controlar o argumento que maximiza a Equação (35). Este controle é realizado aproveitando a variável ψ, que armazena para cada instante de tempo “t” o estado “i”, que produziu a melhor pontuação $\delta_{t-1}(i)a_{ij}$ (RABINER, 1989).

- Terminação:

Melhor pontuação: $$P^* = \underset{1<i<N}{max} \; [\delta_T(i)] \tag{37}$$

Início da sequência oculta: $$q_T^* = \underset{1<i<N}{arg\,max} \; [\delta_T(i)] \tag{38}$$

Neste passo, obtém-se a pontuação P^* (37) da sequência de estados ótima e também o último estado oculto q_T^* (38), pertencente a esta sequência.

- Recuperação da sequência oculta (*path backtracking*):

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, T-3, \dots, 1 \tag{39}$$

Neste passo, a sequência de estados ocultos é recuperada a partir do último estado oculto q_T^*, obtido com o uso da Equação (38) (RABINER, 1989).

5 DESENVOLVIMENTO DO RECONHECEDOR DE FALA

Conforme a formulação estatística descrita nos capítulos anteriores, o vocabulário de um sistema de reconhecimento de fala pode ser representado por um conjunto adequado de HMMs.

O modelo acústico desenvolvido neste trabalho envolveu uma etapa inicial de preparação dos dados de treinamento. Tal etapa consistiu em uma série de passos que precederam a construção e o treinamento do modelo acústico.

5.1 PREPARAÇÃO DOS ARQUIVOS DE ÁUDIO

A preparação dos dados de treinamento requereu a elaboração de um banco de pronúncias. Neste trabalho, foi utilizado um banco constituído pelos dígitos “0”, “1”, “2”, “3”, “4”, “5”, “6”, “7”, “8” e “9”, falados integralmente e previamente gravados por 22 locutores. No preparo do banco, a taxa de amostragem de todos os arquivos de áudio foi convertida de 44100 Hz para 16000 Hz.

Para reforçar as pronúncias de determinadas unidades sonoras, foram realizadas outras gravações com 4 locutores e então as seguintes palavras foram adicionadas ao banco de pronúncias do reconhecedor:

- TITO, fone – [i], para reforçar a pronúncia do digito “7”;
- TINTO, fone – [i~], para reforçar a pronúncia do digito “5”;
- ZEPELIM, fones – [z] e [i~], para reforçar as pronúncias dos dígitos “0” e “5”;
- SEITA, fone – [j], para reforçar as pronúncias de “2”, “3” e “6”;
- MUNDO, fone – [u~], reforçar a pronúncia do digito “1”;
- QUADRAS, fones – [k] e [w], reforçar as pronúncias dos dígitos “4” e “5”;
- NÓ, fone – [ó], reforçar a pronúncia do digito “9”;
- NOBRE, fones – [ó] e [e], reforçar as pronúncia de “3”, “6” e “9”.

Assim como para os dígitos, as palavras listadas também foram gravadas integralmente, utilizando um microfone SKP Podcast-100, usando taxa de amostra-

gem de 16000 Hz e formato em 16-bit. Tais palavras foram escolhidas porque possuem fones em comum com os dígitos, sendo aproveitadas apenas na fase de treinamento dos HMMs e não como parte do vocabulário do sistema implementado. A inclusão destes fones permitiu melhorar o desempenho do reconhecedor balanceando foneticamente o vocabulário. No total, foram gravados 228 arquivos de dígitos e 32 arquivos de reforço.

À medida que a quantidade e qualidade dos dados de treinamento foram sendo ampliadas, foi possível perceber uma melhora significativa no comportamento do reconhecedor, obtendo um sistema mais preciso e capaz de identificar a fala de diferentes usuários.

5.2 OS DADOS DE TREINAMENTO

Os arquivos de áudio, descritos no item anterior, foram utilizados para gerar dados de treinamento para o modelamento do vocabulário. Assim, a informação sonora armazenada em cada arquivo foi codificada em um formato adequado, compatível com o HTK. Para parametrizar os arquivos de áudio foram usados coeficientes mel-cepstrais, convertendo-os do formato PCM para o formato MFCC, conforme descrito no item a seguir.

5.2.1 Parametrização dos Arquivos de Áudio

Inicialmente, o conteúdo de cada arquivo de áudio foi convertido em uma sequência de segmentos utilizando janelas com 25 ms de duração, sendo deslocadas no tempo com um período de atualização de 10 ms. Neste método, foram utilizadas janelas de Hamming para atenuar degradações nas extremidades de cada segmento. O filtro de pré-ênfase foi configurado com o valor 0,97, tipicamente usado em trabalhos de reconhecimento de fala. A função da cóclea foi modelada utilizando um

banco de filtros com 26 filtros triangulares dispostos ao longo da escala Mel, como o ilustrado na Figura 7.

Para formar os vetores de parâmetros, foram extraídos de cada janela do sinal 12 coeficientes mel-cepstrais estáticos (Equação 7), uma componente de energia (Equação 8), 13 componentes de velocidade (Equação 9) e 13 componentes de aceleração (Equação 10). As configurações descritas estabeleceram uma taxa de 100 vetores com 39 parâmetros gerados a partir dos arquivos de áudio a cada segundo.

Os arquivos codificados foram então armazenados em um diretório apropriado, no formato MFCC com compressão. Este diretório foi utilizado como banco de dados de treinamento no desenvolvimento do modelo acústico.

5.3 DESENVOLVIMENTO DO MODELO ACÚSTICO

O modelo acústico consistiu em um conjunto de HMMs devidamente treinados para representar com precisão distribuições de probabilidades associadas às unidades sonoras, nos diferentes contextos nos quais puderam ser identificadas pelo reconhecedor (YOUNG, 1996).

Para modelar o vocabulário, primeiramente cada palavra foi convertida em uma sequência de unidades denominadas fones. Para representar cada fone foi escolhida uma topologia de HMMs *left-right*, os quais foram conectados entre si com a finalidade de formar modelos composto para simular um conjunto hipotético de pronúncias (YOUNG, 1996).

O processo descrito foi desenvolvido com o uso de um dicionário de pronúncias. A construção do dicionário baseou-se em uma lista com a transcrição fonética do conteúdo presente no vocabulário e levou em consideração diferentes tipos de pronúncias de uma mesma palavra. Por fim, foram listados em um arquivo separado todos os monofones presentes em tal dicionário.

5.3.1 O Modelo Protótipo

O modelo protótipo foi utilizado para configurar a topologia de HMM escolhida. Primeiramente, foi especificado o formato dos parâmetros usados na etapa de codificação dos arquivos de áudio (MFCC). Também foram especificados o número de estados e a dimensão da matriz de transição usada em cada modelo.

Com a finalidade de agregar parâmetros de observação, componentes Gaussianas foram associadas aos estados emissores dos modelos protótipos, anexando um vetor com 39 dimensões (μ_{jm}) e uma matriz com dimensão 39x39 (Σ_{jm}) diagonalizada.

Inicialmente os valores das dimensões do vetor, μ_{jm}, e da matriz diagonal, Σ_{jm}, não foram importantes, só interessando para o HTK as dimensões das estruturas a serem utilizadas para o posterior treinamento. Em síntese, para cada fone presente no dicionário de pronúncias e para o silêncio, foram usados HMMs *left-right* com três estados emissores. Como exemplo, a Figura 17 mostra tal topologia de HMM para o fone [u~].

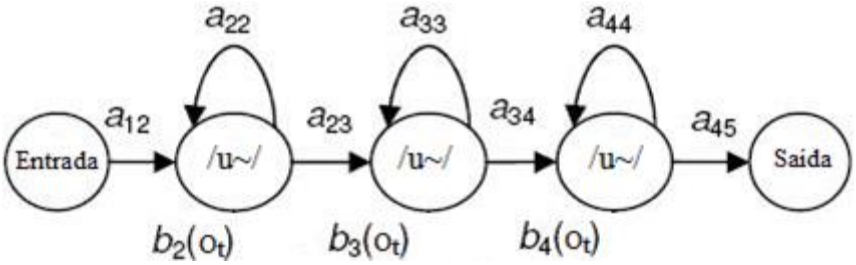


Figura 17 - Topologia *lef-right* para monofones.
Fonte: Adaptado de Gales e Young (2007).

Então, cada palavra presente no vocabulário foi sintetizada por uma sequência de HMMs baseados em fones, conectados conforme especificado no dicionário fonético. Utilizando para cada fone da pronúncia um HMM com a topologia ilustrada na Figura 17 (YOUNG, 1996).

5.3.2 Treinamento dos Modelos

O processo de treinamento foi inicializado atribuindo a cada dimensão dos parâmetros acústicos (μ_{jm} e Σ_{jm}) valores globais de média e variância e configurando as probabilidades de transição (a_{ij}) para serem inicialmente iguais (*flat-start model*) (YOUNG, 2008). No método, todos os modelos protótipos foram inicializados com os mesmos valores globais de médias e variâncias.

Posteriormente, os parâmetros foram re-estimados simultaneamente com o uso dos algoritmos de *forward* e *backward* (Equações 26, 27, 28, 30 e 31) utilizando como dados de treinamento o conjunto de arquivos MFCC. As equações (26), (27) e (28) foram usadas para estimar parâmetros de transição entre os estados (a_{ij}). As equações (30) e (31) foram utilizadas na estimação dos parâmetros de observação (μ_{jm}) e (Σ_{jm}), considerando inicialmente uma componente Gaussiana singular para cada estado emissor.

Os modelos foram treinados executando séries com 3 ciclos de re-estimação. Ciclos de re-estimação excessivos tornam os modelos estreitamente alinhados com os dados de treinamento, deixando de generalizar dados de outros locutores (YOUNG et al., 2006). Para cada ciclo de re-estimação executado, foi gerado um novo conjunto de modelos com parâmetros mais precisos. Também foi gerada uma medida do alinhamento dos novos modelos com os dados disponíveis no banco de treinamento. Os modelos foram re-estimados conforme descrito a seguir.

Primeiro, foram executados três ciclos de re-estimação a partir dos modelos protótipos (Equações 26, 27, 28, 30 e 31). Então, as distribuições de probabilidade foram modificadas para funções baseadas em misturas Gaussianas, com o uso de 2 componentes Gaussianas para cada estado emissor.

Posteriormente, outros 3 ciclos de re-estimação foram executados, novamente com o uso das variáveis de *forward* e *backward*, estimando 1 coeficiente de ponderação para cada componente Gaussiana da mistura.

Em seguida, os modelos foram realinhados com os dados de treinamento utilizando a ferramenta do HTK que executa o algoritmo de Viterbi. No último passo, foi realizada outra série com 3 ciclos de re-estimação usando os modelos realinhados,

resultando no modelo acústico utilizado no sistema de reconhecimento de dígitos do protótipo implementado.

5.4 SISTEMA JULIUS

O modelo acústico, descrito no item anterior, foi desenvolvido para ser executado pelo sistema Julius. Julius é um decodificador de alto desempenho que executa o reconhecimento da fala baseado em um arquivo de gramática, um dicionário fonético e um modelo acústico. Apesar de Julius suportar modelos acústicos construídos no sistema HTK, ele usa um formato próprio de gramática (LEE, 2010).

5.4.1 Desenvolvimento da Gramática

O propósito da gramática foi descrever o padrão de entrada esperado para o sistema de controle da fechadura. A Figura 18 mostra esse padrão, no qual cada palavra é precedida e sucedida por um HMM especial, treinado especificamente para representar o silêncio.

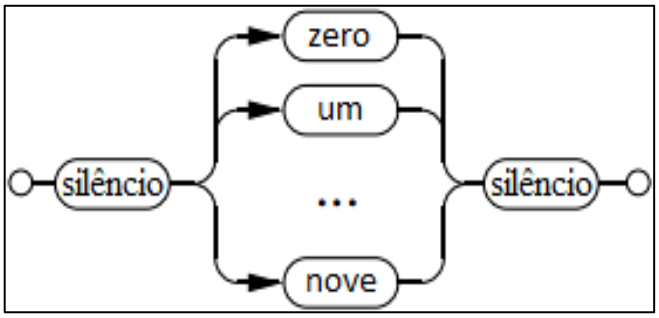


Figura 18 - Padrão gramatical para reconhecimento de dígitos.
Fonte: Young et al. (2006).

Para cada entrada de voz, Julius executa a decodificação respeitando o padrão gramatical definido na Figura 18. Na aplicação, os dígitos foram alocados em uma categoria de palavras candidatas para serem ouvidas ou identificadas pelo decodificador. Quando uma palavra ou sentença de palavras foram ouvidas, Julius ge-

rou um padrão de saída e o resultado da decodificação foi transcrito no LXterminal do Raspberry Pi.

A gramática foi elaborada em dois arquivos separados, então estes arquivos foram compilados para um formato compatível com o decodificador. Para executar o sistema Julius no Raspberry Pi foi utilizado um arquivo de configuração, no qual foram especificadas informações sobre o sistema de reconhecimento implementado e os diretórios onde foram armazenados o modelo acústico, a lista de monofones e a gramática.

6 IMPLEMENTAÇÃO DO PROTÓTIPO

Para implementação do protótipo foi utilizada a placa Raspberry Pi 2, modelo B, que é um microcomputador de baixo custo. O Raspberry Pi (RPi) possui um CI Broadcom que integra componentes de um computador em um único chip (*System-on-Chip*). A placa possui as dimensões de um cartão de crédito e seguintes especificações (MONK, 2013):

- Chip: Broadcom BCM 2836;
- Portas USB: 4;
- Pinos GPIO;
- Saída de vídeo via HDMI;
- Slot para cartão Micro SD.

A Figura 19 mostra o mapa de pinos do RPi e o diagrama esquemático do protótipo implementado.

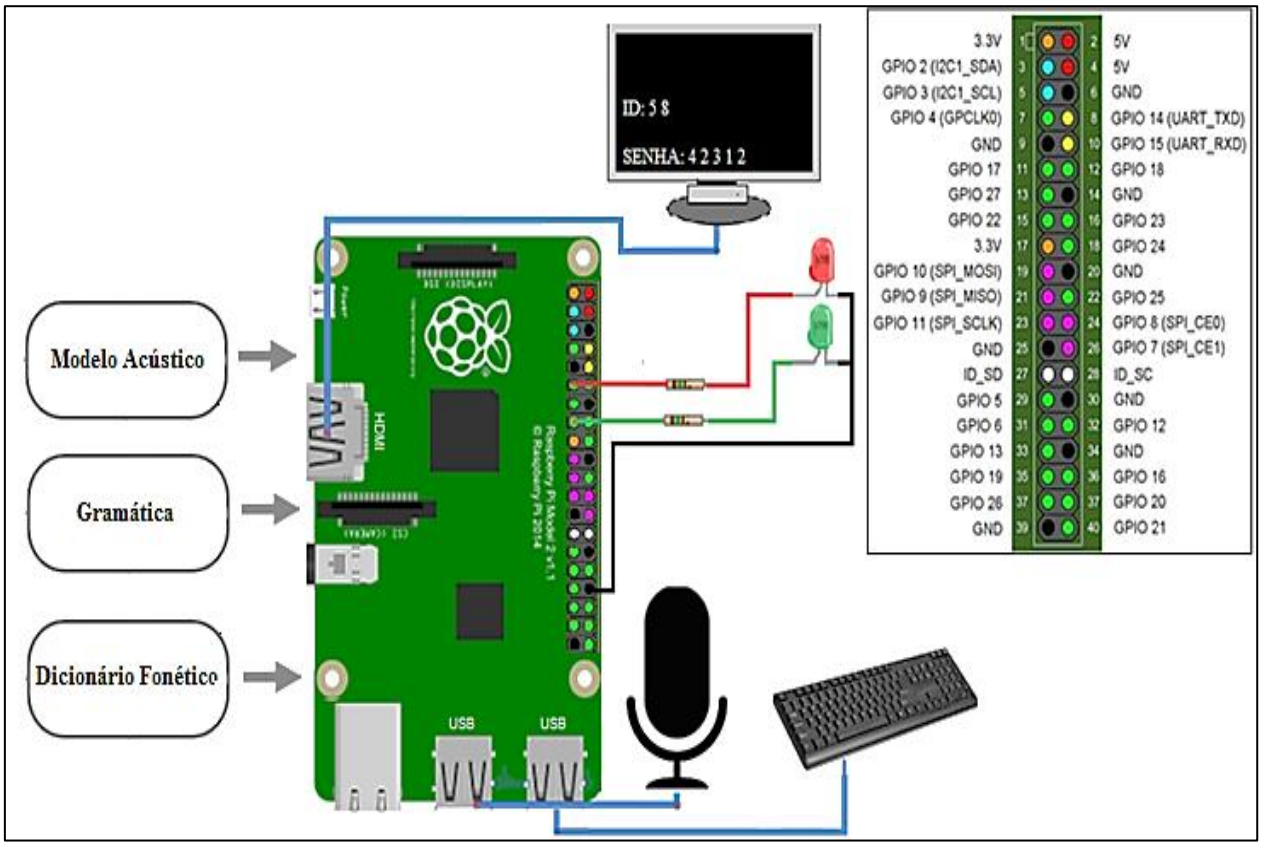


Figura 19 - Diagrama esquamático para o protótipo implementado.

Fonte: Adaptado de Ramkitech (2016).

Além do RPi, foram utilizados para montagem do protótipo os seguintes componentes:

- Microfone SKP Podcast-100;
- Placa de som USB Directsound 3d 7.1;
- Teclado USB;
- LEDs verde e vermelho;
- Resistores de 150 ohms.

A Figura 20 ilustra o protótipo implementado com os componentes listados acima.

Figura 20 - Protótipo implementado.
Fonte: Autoria própria.

O RPi não possui um disco rígido integrado na placa. Por este motivo, foi preparado um cartão micro SD, usando o gerenciador de instalação NOOBS para instalar o sistema operacional Raspbian, recomendado pela fundação Raspberry Pi (MONK, 2013).

Para o reconhecimento da fala foi instalado o sistema Julius no RPi. Para executá-lo foram usados um modelo acústico, um dicionário fonético e uma gramáti-

ca, todos previamente treinados em computador externo, usando o sistema HTK. O RPi usado não possui entrada de áudio integrada. Para acoplar um microfone no RPi foi utilizada uma placa de som USB externa compatível com o Linux.

Na aplicação desenvolvida, o sistema de controle de usuários foi desenvolvido com o uso de LEDs verde e vermelho, para indicar respectivamente a validação (abertura da fechadura) e não validação do usuário.

Conforme o diagrama esquemático (Figura 19), os LEDs foram conectados às portas GPIO do RPi através dos pinos n° 11 (GPIO 17) e n° 15 (GPIO 22) da placa, sendo utilizado o pino n° 34 como GND. No RPi os pinos de propósito geral usam nível lógico baseado em 3,3 volts, para o acionamento dos LEDs foram calculados resistores limitadores de corrente de 150 ohms.

6.1 SISTEMA DE CONTROLE DA FECHADURA

O sistema de controle da fechadura eletrônica (APÊNDICE B) foi implementado em linguagem Python utilizando o interpretador 3.4.3, versão pré-instalada no sistema Raspbian utilizado.

A interação do sistema de controle da fechadura com o sistema Julius foi automatizada com o uso do pexpect que é um modulo Python de correspondências utilizado para controle e automatização de programas interativos. Pexpect foi utilizado para automatizar o envio de comandos ao sistema Julius e o monitoramento dos padrões de saída recebidos do decodificador (SPURRIER, 2015).

Sinteticamente, o funcionamento do sistema de controle da fechadura baseia-se em um cadastro prévio de um código numérico de identificação do usuário e sua senha.

A interface inicial (APÊNDICE C) fornece ao usuário três opções que devem ser acessadas por comando de voz. Ao falar o digito “2” o usuário é direcionado para a interface de cadastramento (APÊNDICE D), na qual o sistema solicita que o usuário digite um número de identificação (ID) com 2 dígitos e uma senha com 5 dígitos

(APÊNDICE E). Depois do cadastro o sistema retorna para a interface inicial, permitindo o cadastro de um novo usuário ou a validação de usuário existente.

Para validar ID e senha o usuário deve falar o digito “3” na interface inicial. Na interface de validação, primeiro o sistema solicita que o usuário fale o ID (APÊNDICE F) e sendo ele válido (existente), o programa solicita que o usuário fale os dígitos referentes à senha previamente cadastrada (APÊNDICE G). Estando a senha correta, o sistema imprime uma mensagem de validação e o LED verde é acionado, conforme Figura 21. Caso ID ou senha estejam incorretos o sistema imprime a mensagem de não validação e o LED vermelho é acionado, como mostra a Figura 22.

Figura 21 - Validação do usuário indicado pelo acionamento do LED verde.
Fonte: Autoria própria.

Figura 22 - Não validação indicada pelo acionamento do LED vermelho.
Fonte: Autoria própria.

6.2 FLUXOGRAMA DO SISTEMA DE CONTROLE DE USUÁRIOS

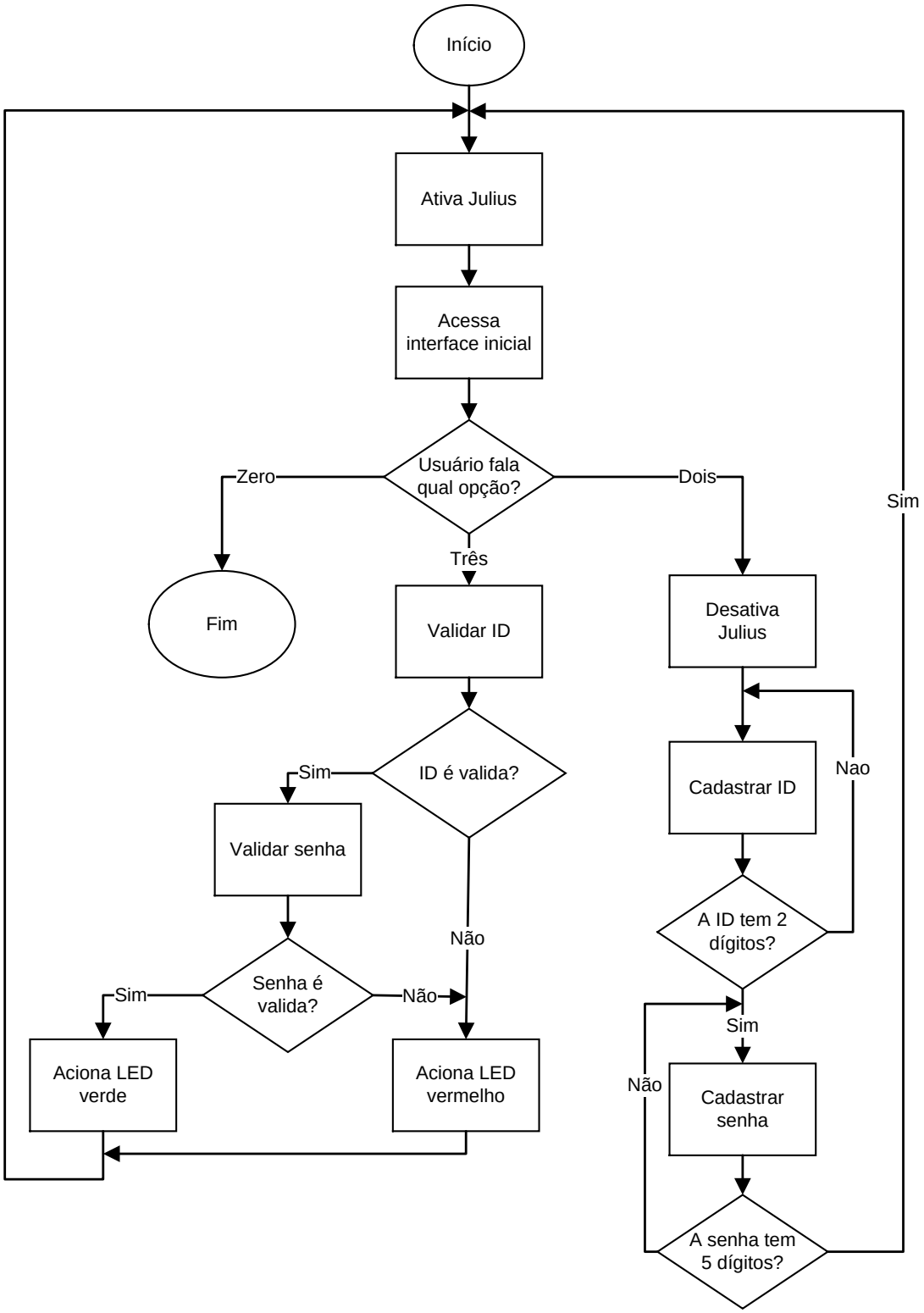


Figura 23 - Fluxograma do sistema de controle de usuários.

Fonte: Autoria própria.

7 CONSIDERAÇÕES FINAIS

Diante do exposto foi possível observar que as tecnologias de reconhecimento de fala serão cada vez mais importantes para o dia a dia das pessoas e continuarão sendo motivo de pesquisas no mundo todo.

O objetivo macro deste trabalho foi modelar um sistema de reconhecimento de fala discreta com independência de locutor para aplicação em fechadura eletrônica usando o sistema de reconhecimento de fala Julius, em conjunto com o sistema de treinamento HTK, e em seguida, implementar um protótipo com o uso do Raspberry Pi para validação do sistema.

Todos os objetivos propostos foram alcançados, o sistema de reconhecimento de fala apresentou funcionamento satisfatório e pode ser validado para diferentes locutores no protótipo implementado com o RPi. Além disso, foi possível compreender os pontos fundamentais da teoria de reconhecimento de fala, das ferramentas de treinamento do HTK e do decodificador de fala Julius.

Nestas condições, observou-se que resultados melhores foram obtidos à medida que a quantidade e a qualidade dos dados de treinamento foram sendo ampliadas. Assim, o sistema de reconhecimento de fala implementado poderá ser aperfeiçoado com a adição de um número maior de gravações de outros locutores ao banco de dados de treinamento, melhorando a independência de locutor.

Por ser um tema de grande importância e muito atual, o conhecimento adquirido poderá trazer bons resultados também para a vida profissional. Ao mesmo tempo, o trabalho ensinou a utilizar na prática as diversas áreas do conhecimento que foram envolvidas em seu desenvolvimento.

Tendo como ponto de partida os resultados obtidos é possível dar continuidade à pesquisa em futuros trabalhos, aperfeiçoando o sistema de reconhecimento de dígitos para outras aplicações como acesso ao *internet banking*, máquinas, equipamentos industriais e domésticos que envolvam o uso de dígitos para seu funcionamento.

Outra sugestão para trabalhos futuros é o desenvolvimento de um sistema de reconhecimento de fala utilizando outras palavras para implementação de diferentes protótipos com o RPi.

Além do HTK existem outros sistemas que podem ser utilizados para o reconhecimento da fala como, por exemplo, o Sphinx e o Matlab. Além dos coeficientes mel-cepstrais (MFCC), utilizados neste trabalho, o HTK permite o uso de outras características, como por exemplo, parâmetros LCP (*Linear Prediction Coefficients*), baseados em modelos de produção da fala e que podem ser extraídas do sinal para uso em trabalhos futuros (MESEGUER, 2009).

Também como sugestão para continuidade da pesquisa, poderá ser desenvolvida uma fechadura eletrônica com reconhecimento de locutor, capaz de identificar o usuário sem a necessidade do cadastramento do código de identificação.

REFERÊNCIAS

ANDREÃO, Rodrigo V. **Implementação em tempo real de um sistema de reconhecimento de dígitos conectados.** 2001. 73 f. Dissertação (Mestre em Engenha- ria Elétrica) – Faculdade de Engenharia Elétrica e de Computação, Universidade - Estadual de Campinas, Campinas, 2001.

BILMES, Jeff A. **A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models.** International Computer Science Institute. Berkeley, 1998.

DENG, Li; YU, Dong. **Automatic speech recognition: a deep learning approach**. 1 ed. Londres: Springer-Verlag, 2015.

DENG, Li; O'SHAUGHNESSY, Douglas. **Speech processing: a dynamic and optimization-oriented approach**. New York: Marcel Dekker, 2003.

DIAS, Raquel de. S. **Normalização de locutor em sistema de reconhecimento de fala** 2000. 113 f. Dissertação de mestrado – Faculdade de Engenharia Elétrica, Universidade Estadual de Campinas, Campinas, 2000.

GALES, Mark; YOUNG, Steve. **The application of hidden markov models in speech recognition.** Cambridge: Fundations and Trends in Signal Processing, v. 1, n. 3. 2007.

GHAHRAMANI, Zoubin. An Introduction to Hidden Markov Models and Bayesian Networks. In: **Hidden Markov Models:** Aplications in Computer Visions, [S.I.]: World Scientific Publishing Company, 2001, p. 9-42, v. 45.

GUYTON, Arthur, C.; HALL, John. E. **Tratado de fisiologia médica.** 11. ed. Rio de Janeiro: Elsevier, 2006.

HOEHN, Katja; MARIEB, Elaine N. **Anatomia e fisiologia**. 3. ed. Porto Alegre: Artmed, 2009.

HOLMES, John N.; HOLMES, Wendy J. **Speech synthesis and recognition**. 2. ed. Londres: Taylor & Francis, 2001.

HOSN, Chadia N. A. **Conversão grafema-fonema para um sistema de reconhecimento de voz com suporte a grandes vocabulários para o português brasileiro** 2006. 77 f. Dissertação de mestrado – Campus Universitário de Guamá, Universidade Federal do Pará, Belém, 2006.

HUANG, Xuedong; ACERO, Alex; HON, Hsiao-Wuen. **Spoken language processing: a guide to theory, algorithm and system development.**1. ed. New Jersey: Prentice Hall, 2001.

JUANG, B. H.; RABINER, Lawrence R. **Fundamentals of speech recognition**. New Jersey: Prentice Hall, 1993.

JUANG, B. H.; RABINER, Lawrence R. **Hidden Markov models for speech recognition**. Technometrics, v. 33, n. 3, 1991.

JURAFSKY, D.; MARTIN, James H. **Speech and language processing: an introduction to natural language processing, speech recognition, and computational linguistics.** 2. ed. New Jersey: Prentice Hall, 2008.

KENT, Ray D.; READ, Charles. **Análise acústica da fala.** São Paulo: Cortez, 2015.

KNILL, K.; YOUNG, Steve. Hidden Markov models in speech and language processing. In: **Corpus based methods in language and speech processing.** Dordrecht: Kluwer Academic Press, 1997, p. 27-68.

LEE, Akinobu. **The Julius book.** 1.0.3 ed. 4.1.5 ver. Nagoya Institute of Technology. Nagoya, 2010.

LI, Jinyu; GONG, Yifan; HAEB-UMBACH, Reinhold; DENG, Li. **Robust automatic speech recognition: a bridge to practical applications.**1 ed. London: Elsevier, 2015.

LI, Xiaolin; PARIZEAU, Marc; PLAMONDON, Réjean. **Training hidden Markov models with multiple observations - a combinatorial method.** IEEE Transactions on PAMI, v. PAMI-22, n. 4. 2000.

LIMA, Carlos Henrique da Rocha. **Gramática normativa da língua portuguesa.** 49 ed. Rio de Janeiro: José Olympio, 2011.

NEY, H. Hidden Markov models in speech and language processing. In: **Corpus based methods in language and speech processing.** Dordrecht: Kluwer Academic Press, 1997, p. 1-26.

QUATIERI, Thomas F. **Discrete-time speech signal processing.** New Jersey: Prentice Hall, 2002.

MESEGUER, Noelia Alcaraz. **Speech Analysis for Automatic Speech Recognition.** 2009. 76 f. Dissertation for the Degree of Master of Science in Eletronics – Department of Electronics and Telecommunications, Norwegian University of Science and Technology, Trondheim, 2009.

MONK, Simon. **Programando o raspberry pi: primeiros passos com python.** 1 ed. São Paulo: Novatec, 2013.

OLIVEIRA, Marcos P. B. **Verificação automática do locutor, dependente do texto, utilizando sistemas híbridos mlp/hmm** 2001. 77 f. Dissertação de mestrado – Faculdade de Engenharia Elétrica, Instituto Militar de Engenharia, Rio de Janeiro, 2001.

OKUNO, Emico; CALDAS, Iberê L.; CHOW, Cecil. **Física para ciências biológicas e biomédicas.** São Paulo: HARBRA, 1986.

PULKKI, Ville; KARJALAINEN, Matti. **Communication acoustics: an introduction to speech, audio and psychoacoustics**. New York: John Wiley & Sons, 2014.

RABINER, Lawrence R. **A tutorial on hidden Markov models and selected applications in speech recognition.** Proceedings of the IEEE, v. 77, n. 2. 1989, p.257-286.

SCHAFER, Ronald W.; RABINER, Lawrence R. **Introduction to digital speech processing.** Fundations and Trends in Signal Processing, v. 1, n. 2. 2007.

SANTOS, E. M. dos. **Engenharia linguística: tecnologias para apoiar as decisões gerenciais na era da internet.** Rio de Janeiro: E-papers, 2008.

SIGMUND, Milan. **Voice recognition by computer.** Marburg: Tectum Verlag, 2003.

SILVA, Diego Furtado; SOUZA, Vinícius Mourão Alves de; BATISTA, Gustavo Enrique de Almeida Prado Alves. A comparative study between MFCC and LSF coefficients in automatic recognition of isolated digits pronounced in Portugues and English. **Revista Acta Scientarium**, Maringá, v. 35, n. 4, p. 621-628, oct.- dec, 2013. Disponível em: <http://periodicos.uem.br/ojs/index.php/ActaSciTechnol>. Acesso em: 13 mar. 2015.

SILVA, Francisco José Fraga da. **Conversão fala-texto em português do brasil integrando segmentação sub-silábica e vocabulário limitado.** 1999. 116 f. Tese de doutorado – Campo Montenegro, Instituto Tecnológico de Aeronáutica, São José dos Campos, 1999.

SPURRIER, Noah. **Pexpect documentation (release 3.3)**. [S.l.: s.n.], 2015.

SIMÕES, Olmos F. **Verificação automática do locutor, dependente do texto, utilizando sistemas híbridos mlp/hmm.** 1999. 77 f. Dissertação de mestrado – Faculdade de Engenharia Elétrica e Computação, Universidade Estadual de Campinas, Campinas, 1999.

WANG, Y. **Model-based approaches to robust speech recognition in diverse environments.** 2015. 215 f. Dissertation for the Degree of Doctor of Philosophy – Engineering Department Cambridge University (Darwin College), Cambridge University, Cambridge, 2015.

YOUNG, Steve. **Large vocabulary continuous speech recognition.** Cambridge University Engineering Department. Cambridge, 1996.

YOUNG, Steve; EVERMANN, Gunnar; GALES, Mark; HAIN, Thomas; KERSHAW, Dan; LIU, Xunying ; MOORE, Gareth; ODELL, Julian; OLLASON, Dave; POVEY, Dan; VALTCHEV, Valtcho; WOODLAND, Phil. **The HTK book (for HTK Version 3.4).** Cambridge University Engineering Department. Cambridge, 2006.

YOUNG, Steve. HMMs and related speech recognition technologies. In: **Corpus based methods in language and speech processing.** Heidelberg: Springer-Verlag, 2008, p. 539-583.

Ramkitech. Disponível em:
<http://www.ramkitech.com/2015/11/iot-remotely-control-led-in-raspberry.html>
Acesso em: 10 jun. 2016.

APÊNDICE A – MODELAGEM DO RECONHECEDOR DE FALA

#Script gera uma lista ordenada de palavras (wlist) a partir das transcrições dos arquivos de áudio.

perl prompts2wlist prompts wlist

#Gera um dicionário de pronúncias (dict) a partir de um dicionário fonte (dict_fonte) e da lista wlist.

HDMan -m -w wlist -n fonemas -l delog dict dict_fonte

#Gera um arquivo *Master Label* (words.mlf) em nível de palavras com a transcrição dos arquivos de áudio no formato HTK.

perl prompts2mlf words.mlf prompts

#Gera um arquivo Master *Label* em nível de fones (fonemas.mlf) com a transcrição fonética do arquivo words.mlf.

HLEd -l '*' -d dict -i fonemas.mlf mkphones.led words.mlf

#Conversão dos arquivos de áudio do formato PCM para o formato MFCC.

HCopy -T 1 -C code_config -S treinoi.scp

#Ferramenta de inicialização dos modelos protótipos com valores globais de média e variância (*flat-start*). Define um nível de referência para os valores re-estimados.

HCompV -T 1 -C code_conf -f 0.01 -m -S treinoii.scp -M hmm0

prototipo

```
#Re-estimação 1

HERest -T 1 -C code_conf -I fonemas.mlf -t 250.0 150.0 1000.0
\ -S treinoii.scp -H hmm0/macros -H hmm0/hmmdefs -M hmm1 mono-
fones


#Re-estimação 2

HERest -T 1 -C code_conf -I fonemas.mlf -t 250.0 150.0 1000.0
\ -S treinoii.scp -H hmm1/macros -H hmm1/hmmdefs -M hmm2 mono-
fones


#Re-estimação 3

HERest -T 1 -C code_conf -I fonemas.mlf -t 250.0 150.0 1000.0
\ -S treinoii.scp -H hmm2/macros -H hmm2/hmmdefs -M hmm3 mono-
fones


#Modificação da distribuição de probabilidades para modelos de
misturas Gaussianas utilizando o script script_mix.conf.

HHEd -H hmm3/macros -H hmm3/hmmdefs -M hmm4 script_mix.conf
monofones


#Re-estimação 4

HERest -T 1 -C code_conf -I fonemas.mlf -t 250.0 150.0 1000.0\
-S treinoii.scp -H hmm4/macros -H hmm4/hmmdefs -M hmm5 monofo-
nes


#Re-estimação 5

HERest -T 1 -C code_conf -I fonemas.mlf -t 250.0 150.0 1000.0\
-S treinoii.scp -H hmm5/macros -H hmm5/hmmdefs -M hmm6 monofo-
nes
```

```
#Re-estimação 6

HERest -T 1 -C code_conf -I fonemas.mlf -t 250.0 150.0 1000.0\
-S treinoii.scp -H hmm6/macros -H hmm6/hmmdefs -M hmm7 monofo-
nes


#Uso do algoritmo de Viterbi para realinhamento dos modelos. O
Master Label aligned.mlf foi usado nos ciclos de re-estimação
finais.

HVite -T 1 -l '*' -o SWT -b SENT-END -C config -H hmm7/macros -H
hmm7/hmmdefs -i aligned.mlf -m -t 250.0 150.0 1000.0 -y lab -a
-I words.mlf -S treinoii.scp dict monofones


# Re-estimação 7

HERest -T 1 -C code_conf -I aligned.mlf -t 250.0 150.0 1000.0\
-S treinoii.scp -H hmm7/macros -H hmm7/hmmdefs -M hmm8 monofo-
nes


# Re-estimação 8

HERest -T 1 -C code_conf -I aligned.mlf -t 250.0 150.0 1000.0\
-S treinoii.scp -H hmm8/macros -H hmm8/hmmdefs -M hmm9 monofo-
nes


# Re-estimação 9

HERest -T 1 -C code_conf -I aligned.mlf -t 250.0 150.0 1000.0\
-S treinoii.scp -H hmm9/macros -H hmm9/hmmdefs -M hmm10 mono-
fones
```

APÊNDICE B – SISTEMA DE CONTROLE DE USUÁRIOS

```
#biblioteca de temporização
import time as delay

#biblioteca GPIO
import RPi.GPIO as GPIO

#biblioteca para manipulação de strings
import string

#biblioteca para automatização de aplicações
Import  pexpect

#biblioteca para interação com o sistema operacional
import os

#biblioteca com funcionalidades do interpretador python
import sys

#limpa a tela
def cls():
    print('\n'*50)

#inicializa dicionário global
i_dictionary = {}
```

```
    #inicializa a interação com o sistema Julius
    def Julius_init():
        child = pexpect.spawn('julius -input mic -C sam-
plee.jconf')
        #define child como variável global
        global child
        #informa Julius qual dispositivo é o microfone
        os.environ['ALSADEV'] = 'plughw:1,0'

    #configuração das portas GPIO
    def GPIO_init():
        #Desabilita warnings
        GPIO.setwarnings(False)
        #Seleciona a GPIO pelo número do pino na placa
        GPIO.setmode(GPIO.BOARD)
        #Saída no pino 11 da placa
        GPIO.setup(11,GPIO.OUT)
        #Saída no pino 15 da placa
        GPIO.setup(15,GPIO.OUT)

    #função reconhecimento de voz
    def speech_catch():
        #inicializa listas locais
        ii_speech = []
        i_speech = []
        #monitora Julius sem definir limite de tempo
        child.expect('please speak', timeout None)
```

```
    #recebe sentenças decodificadas por Julius
    #armazena sentenças na lista i_text
    ii_text = child.before.decode('utf-8')
    i_text = ii_text.split("\n")
    #busca na lista i_text por sentence1
    for line in i_text:
        if line.find('sentence1')!=-1:
            sentence_out = line
            #formata sentence1 para o código
            text = sentence_out.lstrip("sentence1: ")
            ii_speech = text.split()
            i_speech.append(ii_speech[1])
    #armazena dígitos reconhecidos por Julius em speech
    speech ="".join(i_speech)
    #retorna string de dígitos reconhecidos
    return speech


#cadastro de senha
def password(ID):
    cls()
    print("\n    ##### CADASTRO DE SENHA ##### \n")
    print(">>>>Por favor, use '5' dígitos \n")
    #armazena senha cadastrada em i_password
    i_password = str(input('DIGITE A SENHA : '))
    if len(i_password) == 5:
        #associa VALOR senha a CHAVE ID no dicionário
        i_dictionary[ID] = i_password
```

```
            #retorna a tela inicial com ID e senha cadastrados
            Julius_init()
            loop(i_dictionary)
        else:
            #retorna ao cadastro de senha
            print(' A senha deve conter "5" dígitos!!!!')
            password(ID)


#cadastra ID
def ID():
#Interrompe Julius
child.close()
    cls()
    print("    ##### CADASTRO DE ID ##### \n")
    print("A ID deve conter exatamente 2 dígitos")
    i_ID = str(input('DIGITE A ID :'))
    if len(i_ID) == 2:
        #cadastro de senha
        password(i_ID)
    else:
        print(' A ID DEVE CONTER "2" DIGITOS !!!!')
        #retorna cadastro do ID
        ID()
```

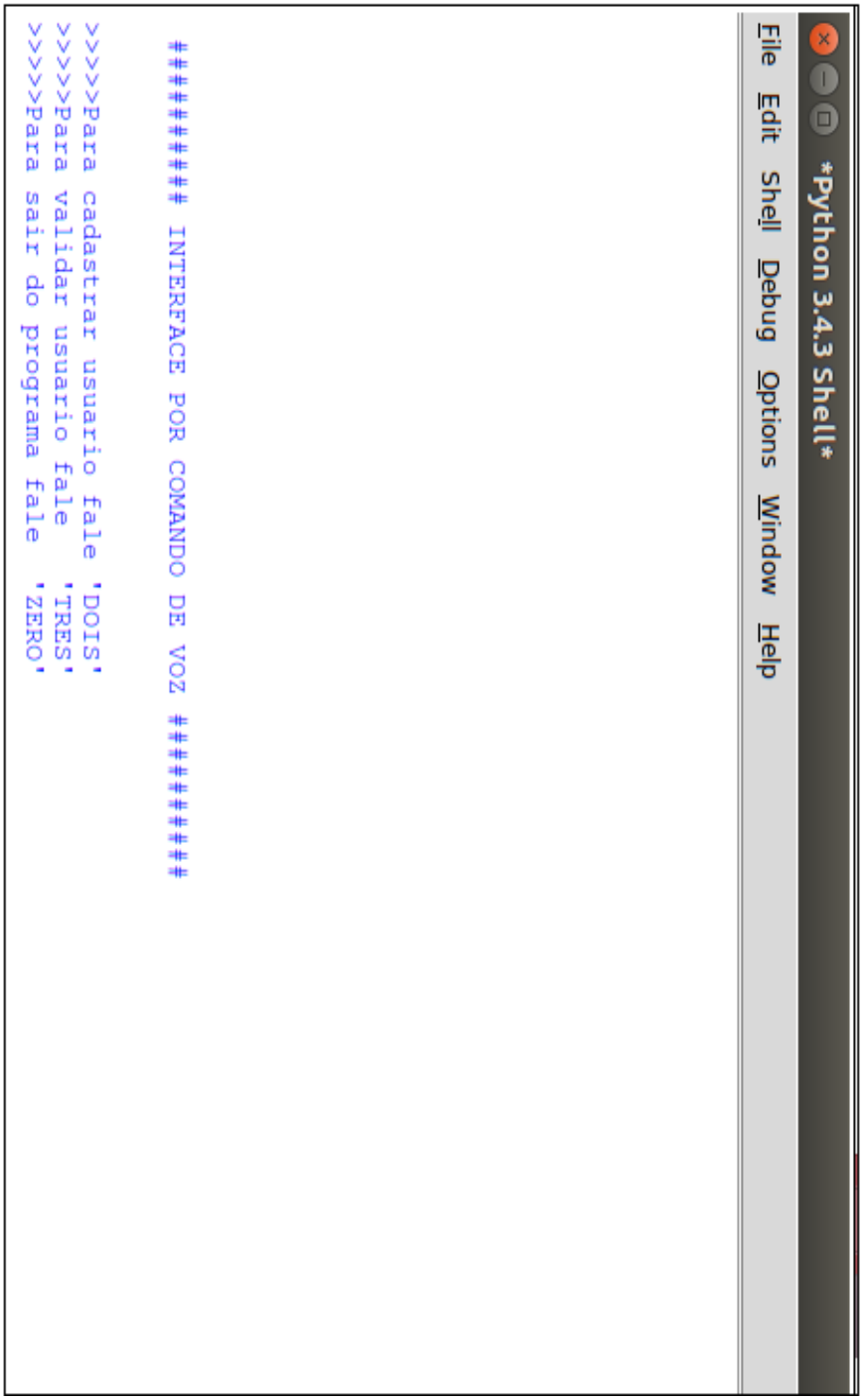
```
#liga LED verde
def LED_verde():
    cls()
    print('>>>>Senha validada com sucesso !!!!')
    GPIO_init()
    #Seta o pino 15 em nível alto
    GPIO.output(15,1)
    delay.sleep(5)
    #Seta o pino 15 em nível baixo
    GPIO.output(15,0)
    delay.sleep(2)
    #limpa GPIO
    GPIO.cleanup()
    #interface inicial
    loop(i_dictionary)

#Liga LED vermelho
def led_vermelho():
    cls()
    print('>>>>>>ID/senha não validada!')
    GPIO_init()
    #Seta pino 11 em nível alto aguarda 5 segundos
    GPIO.output(11,1)
    delay.sleep(5)
    #Seta pino 11 em nível baixo
    GPIO.output(11,0)
    #Espera 2 segundos
```

```
    delay.sleep(2)
    #limpa portas GPIO
    GPIO.cleanup()
    #interface inicial
    loop(i_dictionary)

#valida senha
#recebe senha cadastrada por parâmetro
def password_fala(senha, dicionário):
    #inicializa lista local
    i_fala_password = []
    print('>>>>>>Fale lentamente os digito da senha \n')
    #uso do Julius para capturar a fala do usuário
    for x in range(0,5):
        i_fala_password.append(speech_catch())
        print(i_fala_password)
    #armazena dígitos falados na string password
    password =''.join(i_fala_password)
    #testa password com senha cadastrada
    if password == senha:
        #valida usuário e aciona LED verde
        led_verde()
    else:
        #não valida usuário e aciona LED vermelho
        led_vermelho()
    Return
```

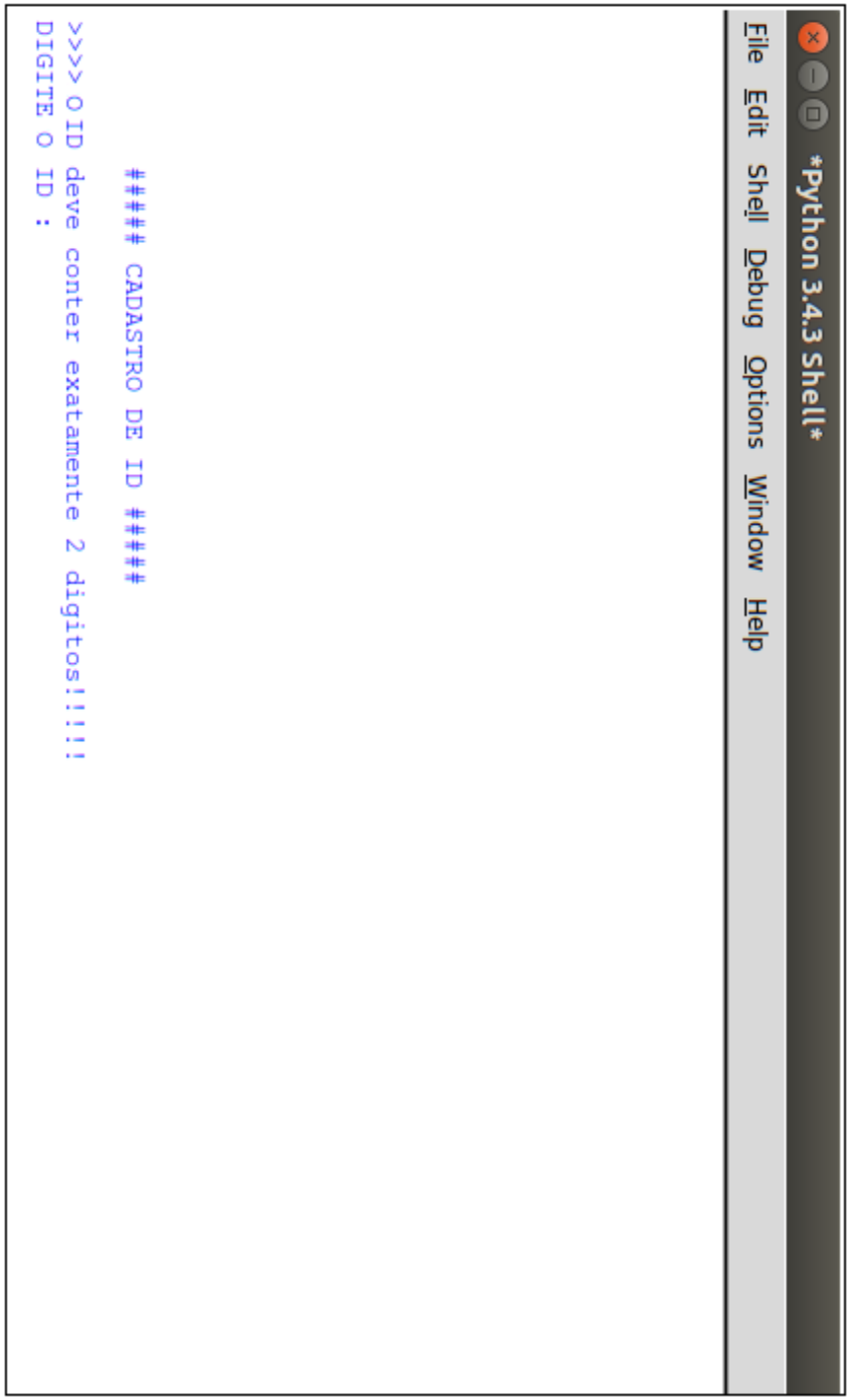
```
#valida ID
def id_fala(i_dictionary):
    cls()
    print('>>>>>>Fale lentamente os digitos da ID \n')
    #inicializa lista local
    i_fala_ID = []
    #uso do Julius para capturar a fala do usuário
    for i in range(0,2):
        i_fala_ID.append(speech_catch())
        print(i_fala_ID)
    #armazena dígitos falados na string fala_ID
    fala_ID = ''.join(i_fala_ID)
    #utiliza fala_ID para verificar a existência do ID
    key = fala_ID
    if key in i_dictionary:
        print('>>>>>>ID validada com sucesso \n')
        #envia VALOR da CHAVE ID para validação
        password_fala(i_dictionary[key], i_dictionary)
    else:
        #não valida e liga LED vermelho
        led_vermelho()
```

```
#interface inicial
def loop(i_dictionary):
    cls()
    print("##### INTERFACE POR COMANDO DE VOZ ######\n")
    print(">>>>>Para cadastrar usuário fale 'DOIS'")
    print(">>>>>Para validar usuário fale   'TRES'")
    print(">>>>>Para sair do programa fale  'ZERO'")
    capture  = '15'

    while True:
        #Aguarda Julius
        capture = speech_catch()

        #cadastra usuário
        if  capture == '2':
            ID()

        #valida usuário
        if  capture == '3':
            id_fala(i_dictionary)

        #encerra aplicação
        if  capture == '0':
            child.close()
            sys.exit()
Julius_init()
loop(i_dictionary)
```

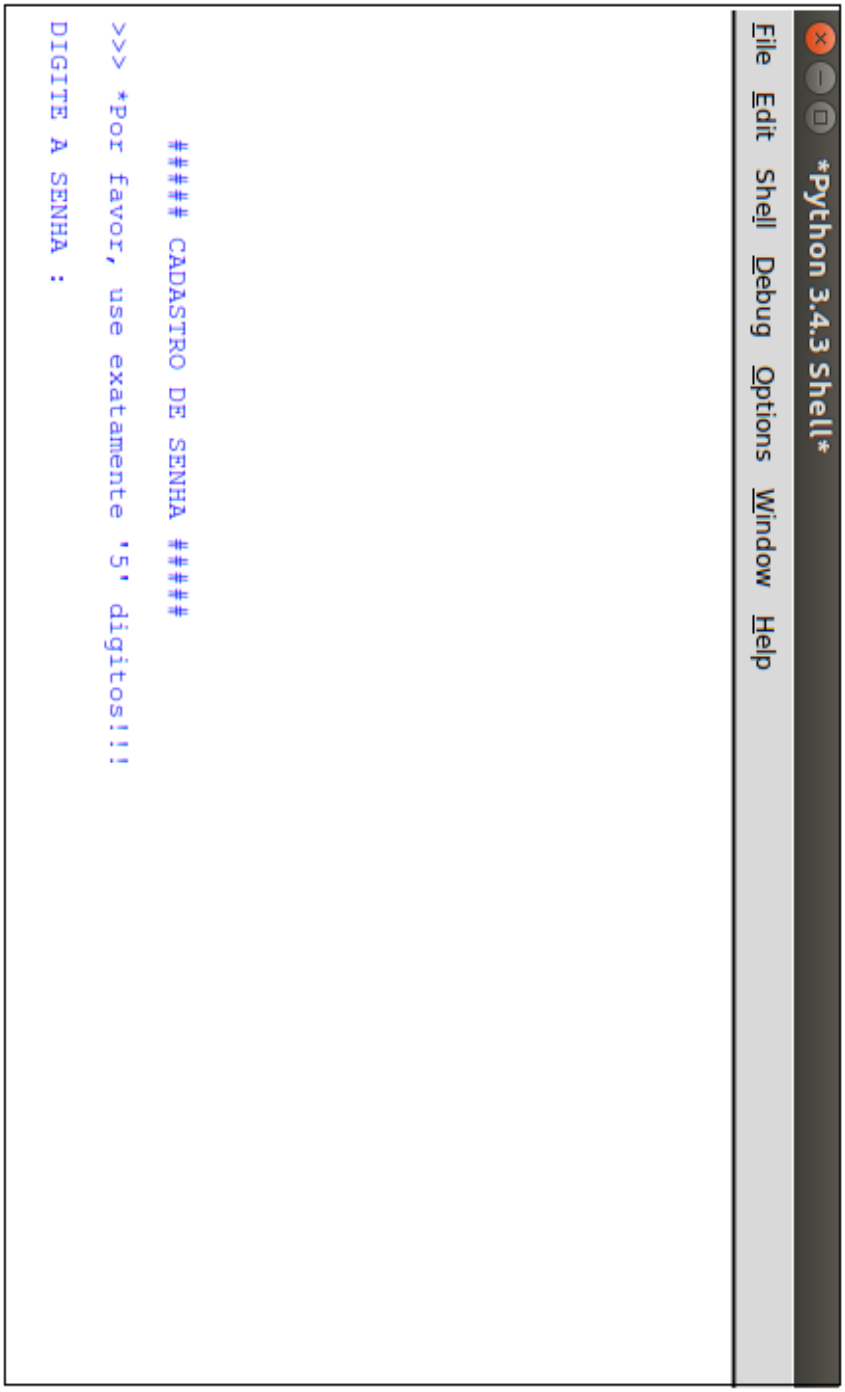
APÊNDICE C – INTERFACE INICIAL



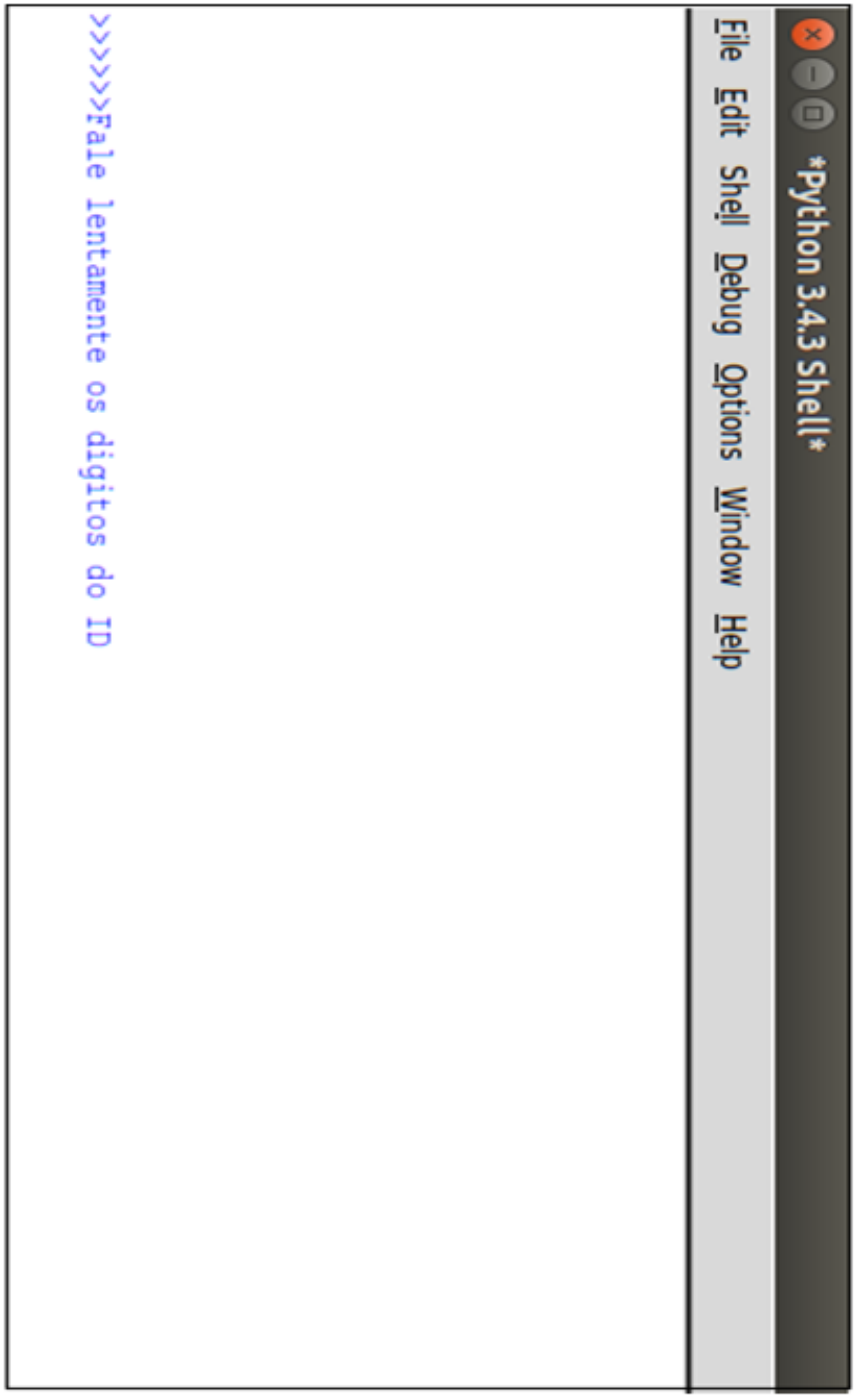
APÊNDICE D – CADASTRAMENTO DO ID



APÊNDICE E – CDASTRAMENTO DE SENHA



APÊNDICE F – VALIDAÇÃO DO ID



APÊNDICE G – VALIDAÇÃO DA SENHA