Universidade Federal do ABC
Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas
Programa de Graduação em Engenharia de Informação

Biometria de Voz: Desenvolvimento de uma Aplicação para Reconhecimento Automático de Locutor

Rodrigo da Silva Cassimiro

Santo André - SP
2022

Rodrigo da Silva Cassimiro

Biometria de Voz: Desenvolvimento de uma Aplicação para Reconhecimento Automático de Locutor

Trabalho de Graduação apresentado ao curso de Engenharia de Informação da Universidade Federal do ABC, como parte dos requisitos necessários para a obtenção do grau de Bacharel em Engenharia de Informação.

Universidade Federal do ABC – UFABC

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas

Programa de Graduação em Engenharia de Informação

Orientador: Prof. Dr. Kenji Nose Filho

Santo André - SP

2022

ATA DE DEFESA DE TRABALHO DE GRADUAÇÃO EM ENGENHARIA DE INFORMAÇÃO

Ata de Defesa do Trabalho de Graduação em Engenharia de Informação da Universidade Federal do ABC

No dia 25 de abril de 2022 reuniu-se a banca examinadora do trabalho apresentado como Trabalho de Graduação em Engenharia de Informação de **Rodrigo da Silva Cassimiro**, intitulado: **"Biometria de Voz: Desenvolvimento de uma Aplicação para Reconhecimento Automático de Locutor"**. Após a exposição oral, o aluno foi arguido pelos componentes da banca que se reuniram reservadamente, e decidiram atribuir o conceito final A.

Prof. Dr. Kenji Nose Filho
Orientador

Prof. Dr. Ricardo Suyama
Avaliador

Me. Tito Caco Curimbaba Spadini
Avaliador

Santo André - SP
2022

Aos meus pais pela base sólida de educação e valores que me orientou durante toda a vida e me conduziu à concretização desta etapa. À minha esposa Cíntia e ao meu filho Murilo pelo amor, paciência, compreensão e incentivos nos momentos de dificultades. Ao meu orientador, sem o qual não teria conseguido concluir esta difícil tarefa.

Resumo

Neste trabalho abordamos o problema da biometria de voz, que consiste no uso de técnicas de reconhecimento de padrões aplicadas às características da voz humana, a fim de identificar automaticamente o indivíduo (locutor). Essa abordagem biométrica tem suas origens nos estudos realizados por Francis Galton, ao final do século XIX, no que diz respeito à identificação do indivíduo através da impressão digital. Ao longo do século XX o desenvolvimento de tecnologias de biometria foi fortemente impulsionado pelas áreas de segurança e ciência forense, culminando, ao final da década de 1980, em sistemas completos capazes de efetuar a leitura e a identificação automática de indivíduos através da impressão digital, bem como, na proposição de sistemas baseados em outras características físicas do indivíduo, como voz, padrão de retina e íris dos olhos. Neste trabalho, a característica física abordada com a finalidade de identificação automática do indivíduo é a voz, a técnica de extração de características de voz é a *Mel-Frequency Cepstral Coefficients* — MFCC, a técnica de reconhecimento de padrões é baseada no *Hidden Markov Models* — HMM, e o banco de dados utilizado para a realização das análises e treinamento do modelo de reconhecimento é o Corpus CEFALA-1: Base de Dados Audiovisual de Locutores para Estudos de Biometria, Fonética e Fonologia, desenvolvida na Universidade Federal de Minas Gerais — UFMG. A aplicação foi desenvolvida através da linguagem de programação Python. Os resultados obtidos são encorajadores, tanto no reconhecimento adequado de um locutor dentre os locutores válidos (locutores cadastrados na aplicação), quanto na identificação de locutores impostores (locutores que não estão cadastrados na aplicação). A média global de acertos do classificador foi de 0,96 (noventa e seis por cento).

Palavras-chaves: Biometria de voz, MFCC, HMM, Mistura de Gaussianas.

Abstract

In this work we deal with the voice biometry problem that consists of measurement and pattern recognition techniques applied to the characteristics of the human voice, to automatically identify an individual (speaker). Biometry has its origins in studies carried out by Francis Galton, at the end of the 19th century, with regard to the identification of the individual through fingerprints. Throughout the 20th century, the development of biometrics technologies was strongly driven by the areas of security and forensic science, culminating, at the end of the 80's, in systems able to perform the automatic reading and identification of individuals using fingerprints, as well as, in the proposition of systems based on other physical characteristics of the individual, for example: voice, retinal pattern and eye iris. In this work, the physical characteristic addressed with the purpose of automatic identification of the individual is the voice, the voice characteristics extraction technique are the *Mel-Frequency Cepstral Coefficients* — MFCC, the pattern recognition is based on *Hidden Markov Models* — HMM, and the database used to perform the analysis and training of the recognition model is Corpus CEFALA-1: Database Audiovisual Speakers for Biometrics, Phonetics and Phonology Studies, developed at the Federal University of Minas Gerais — UFMG. The application was developed using Python programming language. The results obtained are encouraging, both in the proper recognition of a speaker among valid speakers (speakers registered in the application), and in the identification of imposter speakers (speakers who are not registered in the application). The global average of correct answers for the classifier was of 0,96 (ninety six percent).

Keywords: Voice Biometrics, MFCC, HMM, Mixture of Gaussians.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

MFCC	*Mel-frequency Cepstral Coefficients*
LPC	*Linear Predictive Coding*
LPCC	*Linear Prediction Cepstral Coefficients*
HMM	*Hidden Markov Model*
SVM	*Support Vector Machine*
UFMG	Universidade Federal de Minas Gerais
TF	Transformada de Fourier
TDF	Transformada Discreta de Fourier
TDC	Transformada Discreta de Cosseno
f.d.p.	Função Densidade de Probabilidade
GMM	*Gaussian Mixture Model*
HMM-GMM	*Hidden Markov Model-Gaussian Mixture Model, Modelo Oculto de Markov com f.d.p. modelada por mistura de gaussianas*
ASR	*Automatic Speaker Recognition*
UBM	*Universal Background Model*
EM	*Expectation-Maximization*
MAP	*Maximum a Posteriori*
VP	Verdadeiro Positivo
VN	Verdadeiro Negativo
FP	Falso Positivo
FN	Falso Negativo

Sumário

1 Introdução

Por definição, a biometria é a parte da ciência biológica que aplica métodos estatísticos aos seres vivos (BIOMETRIA, 2020). O objetivo desta abordagem consiste, basicamente, na caracterização e mensuração dos indivíduos através de atributos únicos, sejam eles fisiológicos ou comportamentais.

A voz humana, no caso, é um atributo único que carrega informações suficientes para a identificação do locutor. Essa identificação acontece com certo grau de precisão, em função de variações da voz motivadas tanto por fatores físicos quanto por fatores emocionais, bem como em função da técnica empregada para extração das características da voz e da técnica de reconhecimento de padrões utilizada (ALVEZ; CALTABIANO; BOLZAN, 2009).

A problemática da identificação de locutor está fundamentada, basicamente, em três etapas de processamento: a primeira consiste na extração das características a partir de amostras de voz dos locutores de interesse e na montagem de um banco de dados de referência, o qual deve associar a identidade do locutor às suas características de voz; a segunda consiste na utilização da mesma técnica de extração de características da voz, só que desta vez sobre a amostra de voz do locutor, a priori, desconhecido e que se deseja identificar; e a terceira consiste em utilizar técnicas de reconhecimento de padrões para avaliar se a amostra de voz do locutor desconhecido corresponde a alguma amostra de voz cadastrada no banco de dados, respeitando um limiar de semelhança pré-estabelecido.

Algumas das técnicas mais utilizadas para a extração das características de voz são: MFCC, LPC e *Wavelets*. No que diz respeito às técnicas de reconhecimento de padrões aplicadas a esta problemática, podemos citar as redes neurais artificiais, HMM, SVM e técnicas de correlação. Os trabalhos relacionados a seguir utilizam essas técnicas no contexto da problemática da identificação de locutor :

- O trabalho (FECHINE, 1994) utiliza LPC para a extração das características da voz e HMM para o reconhecimento de padrões. Nesse trabalho os resultados são apresentados em termos dos índices de falsa aceitação e falsa rejeição para cada locutor. Entretanto, considerando apenas a quantidade total de acertos em relação ao número total testes realizados, o sistema proposto apresenta uma taxa de acertos de aproximadamente 90,00%.

- O trabalho (ALVEZ; CALTABIANO; BOLZAN, 2009) utiliza *Wavelets* para extração das características da voz e a técnica de correlação linear, através do coeficiente de Pearson, para o reconhecimento de padrões, compondo uma solução de reconheci-

mento automático de locutor com uma taxa de acerto de 93,33%. O referido trabalho cita, ainda, o emprego de redes neurais como uma das técnicas mais utilizadas para reconhecimento de padrões, cujas principais vantagens são a adaptatividade e capacidade de generalização. Entretanto, a mesma é apontada como uma técnica de elevada complexidade de implementação; e

- O trabalho (SILVA; GOMES, 2015) faz o uso de MFCC e SVM para extração das características da voz e reconhecimento de padrões, respectivamente. A aplicação proposta nesse trabalho apresenta uma taxa de acertos de aproximadamente 90,00%.

A identificação de locutor possui uma ampla variedade de aplicações possíveis, dentre as quais se destaca a aplicação em sistemas de autenticação de voz, que tem por objetivo a liberação de acesso mediante a confirmação da identidade do locutor a partir das suas características de voz. Este tipo de autenticação pode ser aplicado, por exemplo, em sistemas de segurança de *smartphones* para desbloqueio da tela.

A Tabela 1 apresenta um comparativo entre as principais técnicas de biometria, relacionando o padrão codificado, a taxa de falhas na identificação, o nível de segurança, o custo, a mudança de característica do padrão codificado ao longo do tempo, a aplicabilidade, a dificuldade para cópia ou roubo e se a técnica é invasiva ou não.

Tabela 1 – Comparativo entre as Técnicas de Biometria. Adaptado de (PINHEIRO, 2019).

Técnica de Biometria	**Padrão Codificado**	**Taxa de Falhas na Identificação**	**Nível de Segurança**	**Custo**
Reconhecimento de voz	Características da voz	1 em 30	Baixo	Baixo
Reconhecimento facial	Perfil, distribuição dos pontos nodais	1 em 100	Baixo	Baixo
Impressão digital	Impressão Digital	1 em 1000	Médio	Baixo
Reconhecimento de íris	Padrões da íris	1 em 1.200.000	Alto	Alto
Reconhecimento de retina	Padrão vascular da retina	(*)	Alto	Alto
	Mudança da Característica	**Aplicabilidade**	**Dificuldade para Cópia ou Roubo**	**Invasiva**
Reconhecimento de voz	Pouco provável	Serviços de telefonia	Baixa	Não
Reconhecimento facial	Muito provável	Instalações de baixa segurança	Alta	Não
Impressão digital	Pouco provável	Autenticação, controle de acesso	Média	Não
Reconhecimento de íris	Improvável	Instalações de alta segurança	Impossível	Sim
Reconhecimento de retina	Improvável	Instalações de alta segurança	Impossível	Sim
(*) Falsa Aceitação da ordem de $0,0001\%$ e Falsa Rejeição da ordem de $0,1\%$. Ref.: EyeDentify ICAM 2001 (http://www.raycosecurity.com/biometrics/EyeDentify.html)				

Analisando os dados da Tabela 1 verifica-se que, dentre as técnicas apresentadas, a técnica de reconhecimento de voz é a que apresenta o menor nível de segurança e maior taxa de falhas no processo de identificação do indivíduo. Entretanto, por se tratar de uma técnica não invasiva, de baixo custo e facilidade de implementação, bem como pelo fato de a voz, em condições normais, sofrer pouca mudança de característica ao longo do tempo, esta é uma técnica que motiva a realização de estudos com a finalidade de melhoria dos níveis de segurança e desempenho, possibilitando a expansão dos nichos de aplicabilidade.

Quanto aos métodos utilizados para a autenticação de voz temos duas abordagens possíveis, a saber: i) autenticação independente do texto, onde não há dependência do conteúdo da fala para a autenticação e o locutor será reconhecido ou autenticado única e exclusivamente em função das suas características de voz; ii) dependente do texto, onde são utilizadas frases-passe padronizadas e, em geral, as mesmas utilizadas na etapa de cadastro do locutor no banco de dados. Ainda em relação ao método dependente do texto, este pode ser implementado no modo estático ou dinâmico. No modo estático, o sistema irá solicitar a mesma frase-passe em todas as autenticações. No modo dinâmico, pode ser utilizada mais de uma frase-passe para a autenticação, inclusive de forma randomizada, desde que cadastrada previamente no banco de dados (AWARE, 2021).

O desenvolvimento de uma aplicação de reconhecimento automático de locutor demanda um banco um banco de dados com quantidade de amostras de áudio suficiente para treinar e testar o modelo de reconhecimento. Neste trabalho, o banco de dados utilizado é o Corpus CEFALA-1: Base de Dados Audiovisual de Locutores para Estudos de Biometria, Fonética e Fonologia, desenvolvida na UFMG (NETO; SILVA; YEHIA, 2019).

No Capítulo 2 faremos uma breve fundamentação teórica, que versará sobre os principais conceitos e aspectos relacionados à forma como o ser humano emite e percebe os sons; captação, digitalização e armazenamento dos sons em formato digital (áudio digital); estruturação do banco de dados dos arquivos de áudio; extração de características da voz a partir de sinais de áudio; reconhecimento de padrões para identificação de locutor; e as métricas de desempenho que serão utilizadas para avaliar o desempenho da aplicação desenvolvida.

No Capítulo 3 serão apresentadas as etapas da metodologia utilizada para o desenvolvimento da aplicação de reconhecimento automático de locutor, bem como a estrutura dos testes idealizados para analisar o desempenho da aplicação.

No Capítulo 4 serão discutidos os aspectos e parâmetros técnicos que influenciam no desempenho da aplicação e apresentados os resultados obtidos.

Por fim, no Capítulo 5 serão apresentadas as conclusões e as perspectivas de trabalhos futuros vislumbradas a partir da realização deste trabalho.

2 Fundamentação teórica

Neste capítulo são abordados os principais conceitos e aspectos relacionados à forma como o ser humano emite e percebe os sons; captação, digitalização e armazenamento dos sons em formato digital (áudio digital); estruturação do banco de dados dos arquivos de áudio; extração de características da voz a partir de sinais de áudio; reconhecimento de padrões para identificação de locutor; e as métricas de desempenho que serão utilizadas para avaliar o desempenho da aplicação desenvolvida. Esse conhecimento prévio tem por objetivo possibilitar a compreensão da proposta de implementação deste trabalho, ou seja, o desenvolvimento de uma aplicação para reconhecimento automático de locutor.

2.1 Emissão e Percepção de Sons

Identificar o locutor através da sua voz requer, minimamente, conhecer os mecanismos de produção da voz, bem como as principais características que diferenciam as vozes de locutores distintos, ou seja, requer obter uma identidade vocal que possa representar um locutor de forma única. Essa identidade vocal é definida, em partes, por características físicas e biológicas do locutor, uma vez que os sons que emitimos são resultados de controles e excitações realizados sobre o chamado aparelho fonador, que é formado por diversos órgãos que compõem o trato vocal, o trato nasal e o sistema sub-glotal, sendo este último considerado a fonte de energia para a produção da voz (FECHINE, 1994).

A produção da voz ocorre pela passagem de ar oriundo dos pulmões pelo trato vocal e pelo trato nasal, acoplados ou não, conectando-se ao ambiente externo na forma de ondas acústicas, cujo espectro de frequências é modelado, sobretudo, em função da seletividade de frequência correspondente à forma modelada pelo trato vocal, uma vez que a forma do trato nasal sofre pouca alteração voluntária por parte do locutor. Nessa abordagem, temos que o trato vocal possui características físicas que variam em função do sexo e da idade do locutor, que são, de certa forma, mais estáveis no tempo, quando comparadas com características tidas como mais dinâmicas, tais como: posição da língua, maxilar e abertura dos lábios, as quais modelam basicamente as dimensões físicas do trato vocal. A Figura 1 apresenta um modelo acústico para o aparelho fonador humano, onde é possível identificar os principais órgãos que o compõem e o seu mecanismo de funcionamento. Basicamente, o conjunto fonador, em função da sua forma, pode ser representado por um conjunto de frequências de ressonância, as quais são chamadas, neste contexto, de frequências formantes, de maneira que as características espectrais da voz dependam diretamente da forma (características dimensionais) do trato vocal, bem como são variantes no tempo (FECHINE, 1994).

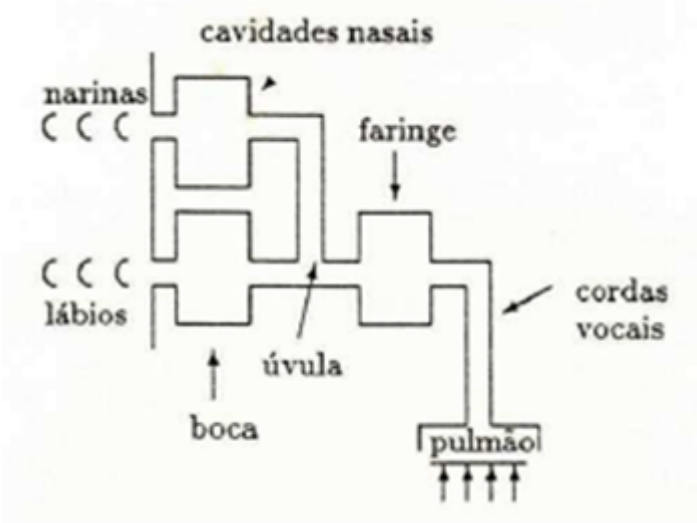


Figura 1 – Modelo acústico para o aparelho fonador humano (FECHINE, 1994).

Face ao exposto, existem basicamente duas formas para se obter uma aproximação que nos permita realizar a identificação automática do locutor, uma consiste em analisar a forma como o locutor fala e a outra através das características inerentes à anatomia do seu trato vocal, sendo esta última menos susceptível a ações de impostores (imitação) e considerada mais estável, embora possa sofrer alterações significativas em função de características de saúde (congestionamento nasal, por exemplo) (FECHINE, 1994). Portanto, sabendo que a caracterização da anatomia do trato vocal pode ser realizada através da análise espectral dos sons produzidos por ele e partindo da premissa que cada indivíduo é único, temos delimitada uma abordagem a ser empregada para a identificação automática do locutor, ou seja, através da análise espectral realizada sobre a sua voz. As questões que ficam em aberto são:

- Quais características da voz são relevantes para a identificação automática do locutor?;
- Como obter essas características?; e
- Como utilizar essas características no processo de identificação do locutor?.

As próximas subseções serão focadas em obter respostas para essas questões. Contudo, um bom candidato a caminho para obter essas respostas é tentar entender o funcionamento do ouvido humano, uma vez que ele é capaz de sentir e decodificar as vibrações ocasionadas pelas ondas sonoras ao nosso redor, as quais compõem um determinado espectro de frequências, entregando ao cérebro as informações acerca dos sons que nos cercam, sejam informações úteis ou somente ruídos ambientes diversos. Essa abordagem será tratada, especificamente, na subseção 2.4, que é dedicada a explicar a técnica de extração de características da voz, a partir de um sinal de áudio correspondente.

2.2 Sinal de Áudio

Um sinal nada mais é do que uma representação de informação acerca de alguma grandeza física do nosso mundo (SEDRA; SMITH KENNETH, 2000). No contexto deste trabalho estamos interessados no sinal que contém informações sobre a voz humana, que consiste em ondas acústicas (ondas de pressão de ar). A este sinal é atribuído o nome de sinal de áudio.

Um sinal é encontrado na natureza no formato analógico, como é o caso das ondas acústicas. Para que seja possível extrair informações desse sinal é preciso convertê-lo em alguma grandeza para a qual tenhamos mais facilidade de processamento. No caso, a forma usual aplicada ao tratamento de sinal de áudio é convertê-lo em grandeza elétrica, na forma de tensão ou corrente elétrica, através de um transdutor de pressão, como o microfone, obtendo uma representação elétrica equivalente do sinal no domínio elétrico, ainda no formato analógico (SEDRA; SMITH KENNETH, 2000).

Para a extração de informações de um sinal é necessário realizar o seu processamento, o que atualmente é realizado por sistemas eletrônicos digitais, demandando uma etapa de conversão do sinal analógico em sinal digital (SEDRA; SMITH KENNETH, 2000). Esta etapa consiste basicamente de um processo de amostragem e quantização, com taxa de amostragem e número específico de bits, resultando em um sinal de áudio no formato digital (SILVA; SERRA, 2019).

Uma característica importante do sinal é que ele pode ser analisado tanto no domínio do tempo quanto no domínio da frequência (análise espectral) de maneira equivalente, sendo esta última representação do sinal obtida através de ferramentas matemáticas como a Transformada de Fourier (TF). No caso, para o processamento digital de sinais em ambiente computacional, como estamos trabalhando com sinais discretos no tempo, aplica-se a ferramenta matemática equivalente, ou seja, a Transformada Discreta de Fourier (TDF) (SILVA; SERRA, 2019).

Na análise espectral é possível visualizar com clareza a faixa de frequência de abrangência do sinal e suas respectivas magnitudes e fases. Em geral, a faixa de sons audíveis à maioria dos seres humanos está compreendida na região de 20 Hz a 20 kHz. O formato de áudio digital utilizado neste trabalho é o WAV, com taxa de amostragem de 44100 Hz e 16 bits de quantização. Esses áudios são do tipo monaural, ou seja, com um único sinal analisado por vez.

2.3 Banco de Dados

Uma das principais premissas da proposta de desenvolvimento deste trabalho está fundamentada na definição da estrutura do banco de dados que seja capaz de possibilitar a

realização do treinamento e a análise de desempenho da aplicação desenvolvida. Portanto, a estrutura idealizada prevê basicamente duas características imprescindíveis. São elas:

- sinais de áudio de curta duração de um mesmo locutor falando diversas frases distintas em conteúdo;
- sinais de áudio de curta duração de diferentes locutores falando as mesmas frases citadas no tópico anterior.

A partir das definições supracitadas iniciou-se a pesquisa em busca de bancos de dados que atendesse aos requisitos estabelecidos, chegando ao Corpus CEFALA-1: Base de Dados Audiovisual de Locutores para Estudos de Biometria, Fonética e Fonologia, desenvolvida na UFMG, que consiste em uma base de dados pública e gratuita que reúne arquivos de áudio digital de 104 locutores, gravados sob o mesmo protocolo de coleta e em ambiente controlado (i.e., estúdio profissional de gravação com as seguintes dimensões: 2,8 m (largura) x 2,9 m (comprimento) x 2,2 m (altura)). Para cada um dos locutores, o mesmo sinal de áudio foi gravado simultaneamente através de 5 equipamentos distintos, gerando 5 arquivos de áudio para cada um deles. Dentre os locutores cadastrados na base de dados, 49 são do sexo feminino e 55 do sexo masculino. As gravações, segundo o protocolo de coleta, foram divididas em três etapas, a saber: i) entrevista composta de um relato pessoal (fato marcante da vida e o que fez nas últimas férias), um comentário sobre o programa de televisão favorito e o motivo da predileção, uma descrição sobre a atividade laboral exercida e sobre o que gosta de fazer no tempo livre, e um relato não pessoal (contar um caso ou fato de conhecimento); ii) leitura do trecho de um livro (HARSANYI, Zsolt. A vida de Galileu: o contemplador de estrelas. Livraria José Olympio, 1957); e iii) leitura de um total de 20 frases. O tempo aproximado de gravação para cada um dos locutores é de 5 minutos. Esse tempo compreende as três etapas do protocolo de coleta (NETO; SILVA; YEHIA, 2019).

2.4 Técnica de Extração de Características do Sinal de Áudio: MFCC

A extração de características da voz, a partir de sinais de áudio, através da aplicação da técnica MFCC é uma das principais técnicas utilizadas em sistemas de reconhecimento automático de locutor. Essa técnica teve sua versão original proposta na década de 1980, por Davis e Mermelstein (DAVIS; MERMELSTEIN, 1980) como uma alternativa aos LPCs e LPCCs. Conforme antecipado na subseção 2.1, esta é uma técnica que leva em consideração a maneira de funcionamento do ouvido humano, ou seja, o seu caráter de percepção não linear, que é aplicado através do cálculo do espectro de frequências do sinal filtrado por um banco de filtros de envelope triangular, cuja largura de banda e

frequências de corte (ou centrais) são definidos de acordo com a escala de frequências Mel (CRIPTOGRAPHY, 2012) (TODOR; NIKOS; GEORGE, 2005).

A implementação de um processamento de áudio através de MFCC para obter vetores de características capazes de representar o locutor consiste em um algoritmo composto, basicamente, pelos seguintes passos, conforme proposto em (CRIPTOGRAPHY, 2012) e (FACHINI; HEINEN, 2016):

- **Passo 1**: Segmentação do sinal através de um processo chamado janelamento, que consiste em dividir o sinal de áudio em segmentos menores, chamados de quadro, de tamanhos iguais e multiplicá-los por uma função de janelamento. Comumente é utilizada a janela de Hamming quando estamos tratando de sinais de áudio (Equação 2.1):

$$w(n) = 0.54 - 0.46\ cos(\frac{2\pi n}{N-1}),\ 0 \leq n \leq N-1 \tag{2.1}$$

  onde N é o tamanho da janela.

  A resultante do janelamento do sinal é obtida através da Equação 2.2.:

$$s_i(n) = s(n + iD)w(n) \tag{2.2}$$

  onde $s_i(n)$ são os sinais resultantes do processo de janelamento (quadros), com uma sobreposição entre as janelas de $\frac{N-D}{N}$, $s(n)$ é o sinal de áudio original e $w(n)$ a janela utilizada. O número de janelas de tamanho $N-1$ é dado por $\frac{L-N}{D}+1$, onde L é o número de amostras do sinal de áudio original.

- **Passo 2**: Aplicação da TDF (Equação 2.3) e obtenção da densidade espectral de potência do sinal para cada um dos quadros (Equação 2.4);

$$S_i(k) = \sum_{n=0}^{N-1} s_i(k)e^{-j2\pi\frac{k\ n}{N}} \tag{2.3}$$

$$P_i(k) = \frac{1}{N}|S_i(k)|^2 \tag{2.4}$$

- **Passo 3**: Filtragem do resultado obtido no passo anterior através de um banco de filtros triangulares cujas frequências de corte são definidas a partir da escala de frequências Mel. Na saída de cada filtro iremos obter o sinal $X_{i,m}(k)$, para $m = 0, ..., M-1$, onde M é a quantidade de filtros que compõem o banco, tipicamente 26 (CRIPTOGRAPHY, 2012). Posteriormente é então calculada a energia de cada sinal $Y_i(m) = \sum_{k=0}^{N-1} X_{i,m}(k)$. A Equação 2.5 apresenta a equivalência entre a escala linear de frequências e a escala de frequências Mel:

$$Pitch(mel) = 1127.0148\ ln(1 + \frac{f(Hz)}{700}) \tag{2.5}$$

- **Passo 4**: Cálculo do logaritmo dos valores obtidos anteriormente (26 para cada quadro) (Equação 2.6);

$$c_i(m) = logY_i(m),\ 0 \leq m \leq M-1 \tag{2.6}$$

- **Passo 5**: Cálculo da TDC, obtendo os Coeficientes Mel-Cepstrais de interesse (Equação 2.7):

$$c_{mel_i}(m) = \sum_{k=0}^{M-1} c_{i,k}\ cos(\frac{\pi}{M}(k+\frac{1}{2})m) \tag{2.7}$$

A Figura 2 apresenta o diagrama em blocos para o algoritmo supracitado. Já a Figura 3 apresenta um detalhamento dos passos 2 e 3 do referido algoritmo, ou seja, considera que o sinal no domínio do tempo corresponde a um quadro resultante do processo de janelamento. Oportunamente, é importante destacar que os passos 2, 3, 4 e 5 do algoritmo são aplicados sobre cada um dos quadros resultantes do processo de janelamento e que cada quadro dará origem a um vetor de 26 coeficientes MFCC.

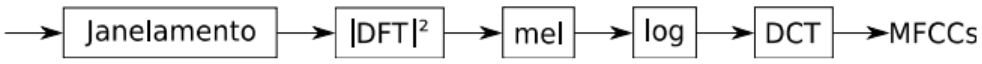


Figura 2 – Diagrama para o Algoritmo MFCC (FACHINI; HEINEN, 2016).

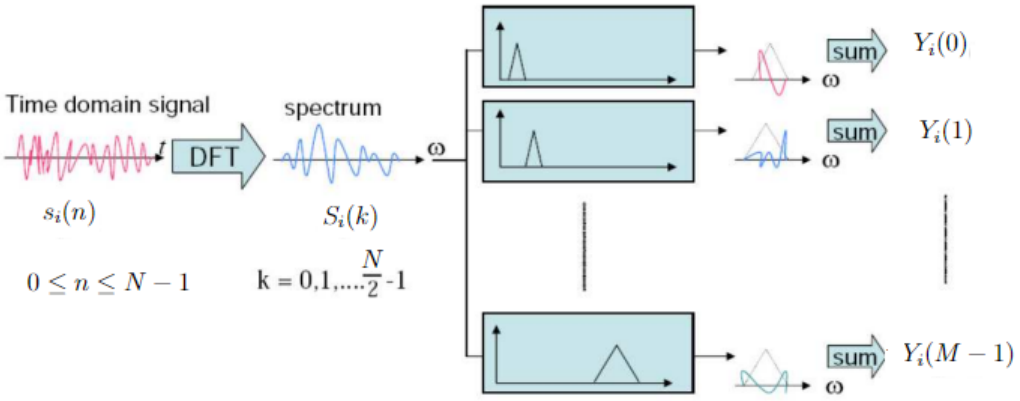


Figura 3 – Diagrama para o Algoritmo MFCC com Detalhamento dos Passos 2 e 3. Adaptado de (JURAFSKY; MARTIN, 2009 apud FACHINI; HEINEN, 2016).

O motivo pelo qual realizamos o processo de janelamento está relacionado ao comportamento do sinal de áudio, que em intervalos curtos de tempo, comumente entre 10 e 30 ms, pode ser considerado estatisticamente estacionário, condição necessária para a aplicação deste tipo de análise. A obtenção da energia do sinal em cada um dos quadros e a tomada do logaritmo dos valores obtidos na saída do banco de filtros são aproximações relacionadas à forma de funcionamento do ouvido humano, ou seja, leva em consideração a não linearidade da percepção dos sons (não percebemos o volume do som em escala

linear) e a dificuldade de distinguir frequências muito próximas. O cálculo da TDC tem por objetivo descorrelacionar os valores para as energias obtidas na saída do banco de filtros, que, *a priori*, por serem filtros sobrepostos, possuem elevada correlação. Sem esta etapa, os coeficientes Mel-Cepstrais não satisfazeriam as condições necessárias para modelar, por exemplo, o classificador HMM. Ressalta-se, ainda, que dos 26 coeficientes obtidos somente 12 serão utilizados para o tipo de análise proposta neste trabalho (coeficientes de 2 a 13), visto que os demais apresentam características de não estacionaridade estatística, o que via de regra pode degradar o desempenho do classificador (CRIPTOGRAPHY, 2012).

2.5 Técnica de Reconhecimento de Padrões: *Hidden Markov Models* - HMM

Um Modelo Oculto de Markov, do inglês *Hidden Markov Model* – HMM, consiste em um modelo estatístico definido por parâmetros, a *priori*, desconhecidos, denominados como parâmetros ocultos, que serão determinados a partir de parâmetros observáveis (símbolos). Os parâmetros que definem um HMM são $\lambda = (A, B, \pi)$ (FECHINE, 1994) e (KUINCHTNER, 2018):

- λ, Modelo HMM;
- N, número de estados (ocultos);
- M, número de observações por estado;
- $O = [O_1, O_2, \ldots, O_t]$, vetor com a sequência de observações;
- $A = [a_{ij}], 1 \leq i, j \leq N$, matriz de transição entre estados;
- $B = [b_j(k)], 1 \leq j \leq N \; e \; 1 \leq k \leq M$, f.d.p. de emissão de símbolos associada a cada um dos estados;
- $\pi = \pi_i, 1 \leq i \leq N$, matriz de probabilidade do estado inicial.

Ao modelar um HMM o número de estados será sempre finito (discretos). As observações, por sua vez, podem ser modeladas através de f.d.p. discretas ou contínuas, a depender do tipo de fenômeno que se deseja modelar. No caso de aplicações de voz, pela sua natureza contínua, em geral, emprega-se f.d.p. contínuas (KUINCHTNER, 2018). Portanto, o HMM empregado neste trabalho utiliza f.d.p. de emissão de símbolos modelada por misturas de gaussianas, ou seja, consiste em um HMM-GMM.

A quantidade de estados ocultos de um HMM está muito mais relacionada com a complexidade matemática e computacional do modelo, do que com a qualidade dos

resultados obtidos (ANDRADE, 2000). No caso deste trabalho o resultado avaliado é a taxa de acerto no reconhecimento do locutor.

Especificamente para o caso da utilização de algoritmos baseados em *Forward & Backward* para os problemas de treinamento e avaliação, a complexidade de cálculo computacional é da ordem de N^2M operações, onde N é o número de estados (ocultos) e M é número de observações (ANDRADE, 2000).

Portanto, elevar o número de estados de um HMM implicará em torná-lo mais lento, em função do maior número de operações matemáticas a serem realizadas. Já a quantidade de observações terá menor impacto na complexidade computacional do HMM. Entretanto, ao contrário do que ocorre com a quantidade de estados, a quantidade de observações implicará de maneira significativa na qualidade dos resultados, ou seja, uma quantidade elevada de observações elevará a complexidade de cálculo computacional, deixando o modelo mais lento, mas uma quantidade pequena de observações tenderá a reduzir o desempenho do modelo drasticamente.

As Figuras 4 e 5 apresentam, respectivamente, um modelo ergódico e um modelo esquerda-direita de HMM, ambos com f.d.p. de emissões de símbolos modelados por misturas de gaussianas. As denominações ergódica e esquerda-direita referem-se aos tipos de transição entre estados de um HMM. No caso do modelo ergódico todos os estados estão conectados e podem ser alcançados a partir de qualquer outro estado do modelo, em função de um número finito de passos. Já no caso do modelo esquerda-direita não é permitido que um estado retroceda a um estado já acessado, ou seja, o algoritmo é sequencial e avança da esquerda para a direita (KUINCHTNER, 2018).

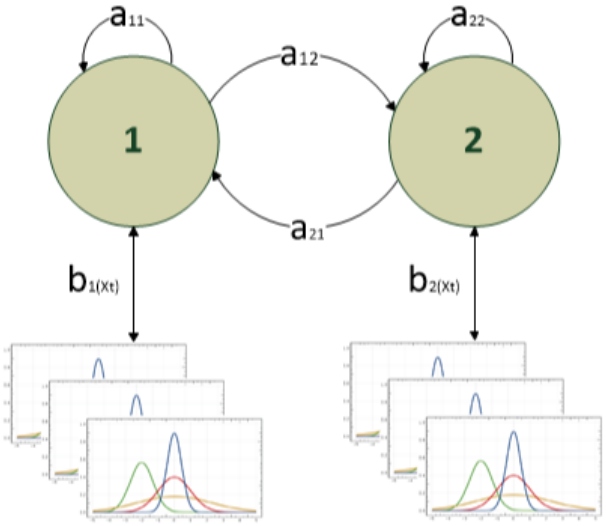


Figura 4 – HMM ergódico com f.d.p de emissão de símbolos contínua. Adaptado de (KUINCHTNER, 2018).

O Processo de Markov presente em ambos modelos é caracterizado pelo fato de que a transição para um novo estado depende apenas do estado corrente. O processo consiste

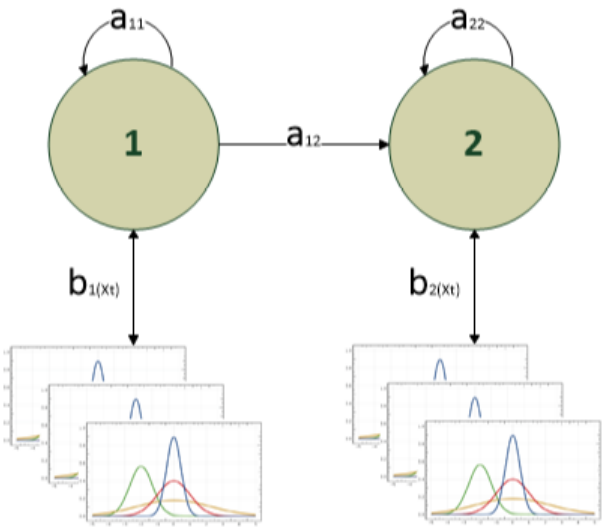


Figura 5 – HMM esquerda-direita com f.d.p de emissão de símbolos contínua. Adaptado de (KUINCHTNER, 2018).

em um algoritmo iterativo que é iniciado em um dos estados do modelo, no instante inicial, a depender da distribuição de probabilidade inicial de estado ($\pi = \pi_i$), produzindo um determinado símbolo, que, por sua vez, dependerá da f.d.p. de emissão de símbolo associado a este estado ($B = [b_j(k)]$). No instante seguinte pode ocorrer a transição para outro estado ou a permanência no estado corrente, a depender da distribuição de probabilidade de transição de estados ($A = [a_{ij}]$) (KUINCHTNER, 2018).

Em linhas gerais, a entrada do HMM são as sequências de observações (símbolos), as quais, através de algoritmos específicos, possibilita a estimação dos parâmetros ocultos do modelo. Os algoritmos de computação dinâmica que permitem a implementação prática de um HMM são empregados no contexto da resolução de três problemas principais, a saber (FECHINE, 1994), (KUINCHTNER, 2018) e (ANDRADE, 2000):

- **Problema de decodificação:** O problema de decodificação consiste em descobrir os parâmetros ocultos do modelo, ou seja, a sequência de estados que melhor representa a sequência de observações. Neste caso, para encontrar a melhor sequência de estados, utiliza-se o algoritmo Viterbi;

- **Problema de Treinamento:** O problema de treinamento consiste em ajustar os parâmetros do modelo com base em uma sequência de observações, empregada com a finalidade de treinamento, de modo a obter os parâmetros ótimos que maximizem a probabilidade do modelo ter produzido tal sequência de observações. Um dos principais algoritmos utilizados para solucionar o problema de treinamento é o *Expectation-Maximization* - EM, através de um caso particular denominado algoritmo de Baum-Welch (baseado no processo *Forward & Backward*);

- **Problema de Avaliação:** O problema de avaliação consiste em avaliar a probabi-

lidade de um determinado modelo (λ) ter emitido uma determinada sequência de observações ($O = [O_1, O_2, \ldots, O_t]$). Para este problema um dos algoritmos que pode ser empregado para solucionar o problema de avaliação é o *Forward & Backward*.

Devida a complexidade envolvida na modelagem de um HMM a maneira mais fácil de demonstrar a sua aplicação é através de um exemplo de menor complexidade, que permita o seu entendimento e, consequentemente, a sua generalização para problemas mais complexos. Portanto, considere um experimento composto por duas moedas viciadas (S_1 e S_2), lançadas uma de cada vez. Sejam os eventos $O_1 = CARA$ e $O_2 = COROA$, assumindo que $P(O_1|S_1) = 2/3$, $P(O_2|S_1) = 1/3$ e $P(O_1|S_2) = 1/6$, $P(O_2|S_2) = 5/6$. Assumindo ainda que as duas moedas possuem a mesma probabilidade de ser a escolhida para iniciar o experimento, e que a cada lançamento ambas as moedas possuem a mesma probabilidade de serem escolhidas, um HMM possível para representar este experimento é o $\lambda = (A, B, \pi)$, representado pela Figura 6, com:

$$A = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix};\ B = \begin{bmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{1}{6} & \frac{5}{6} \end{bmatrix};\ \pi = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

Observe que os elementos da matriz A estão relacionadas às probabilidades de que a cada lançamento ambas as moedas possuem a mesma probabilidade de serem escolhidas, ou seja $a_{11} = P(S_1|S_1) = 1/2$ (Probabilidade de sortear a moeda S_1 no segundo lançamento dado que no primeiro lançamento foi sorteada a moeda S_1), $a_{12} = P(S_2|S_1) = 1/2$, $a_{21} = P(S_1|S_2) = 1/2$ e , $a_{22} = P(S_2|S_2) = 1/2$. Já os elementos $\pi_1 = \pi_2 = 1/2$ está relacionado ao fato de que as duas moedas possuem a mesma probabilidade de ser a escolhida para iniciar o experimento.

Por fim, os elementos da matriz B estão relacionados com as probabilidades de ocorrência dos eventos $O_1 = CARA$ e $O_2 = COROA$, ou seja, $b_{11} = P(O_1|S_1) = 2/3$, $b_{12} = P(O_2|S_1) = 1/3$, $b_{21} = P(O_1|S_2) = 1/6$, $b_{22} = P(O_2|S_2) = 5/6$.

No entanto, pode ser que os parâmetros do modelo (A, B e π) sejam desconhecidos. Neste caso, dado uma série de realizações do experimento, estes parâmetros poderiam ser obtidos através do algoritmo empregado para resolver o "Problema de Treinamento".

Conhecidos ou obtidos os parâmetros do HMM podemos estar interessados em calcular a probabilidade de ocorrência da sequência de observações $O = [O_1, O_2]$. Este é um exemplo do "Problema de Avaliação". Neste caso, o HMM permite as sequências de estados ocultos $Y = [[S_1, S_1], [S_1, S_2], [S_2, S_1], [S_2, S_2]]$, através das quais é possível a obtenção da sequência observada. Para cada uma dessas sequências de estados ocultos é possível calcular a probabilidade associada à emissão de $O = [O_1, O_2]$:

- **Para** $Y_1 = [S_1, S_1]$**:**

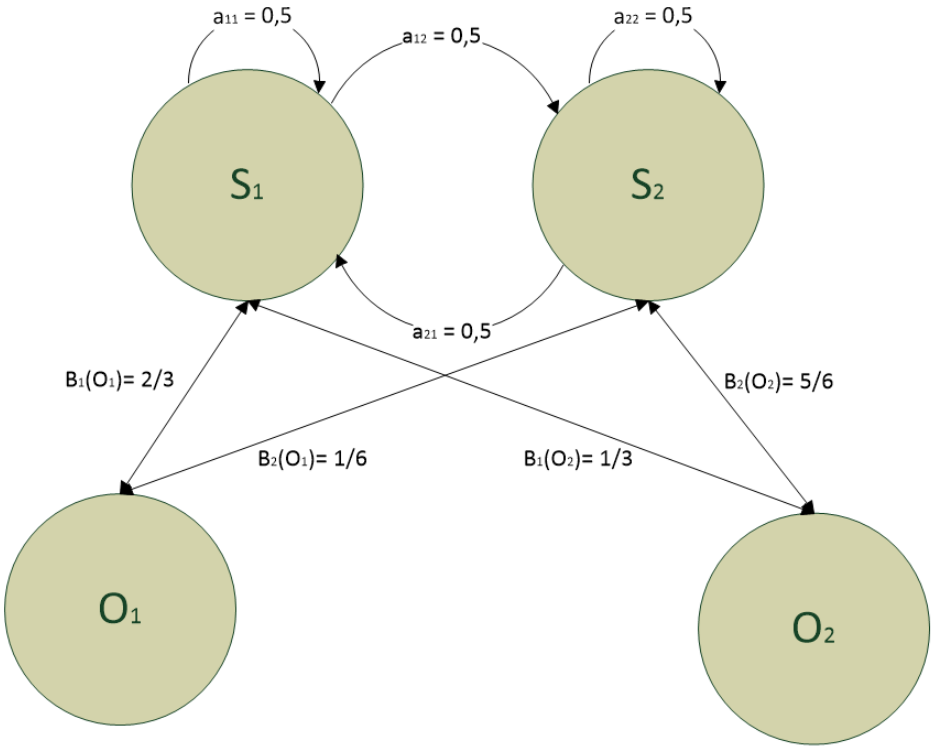


Figura 6 – Exemplo de Aplicação para o HMM - Lançamento de 2 Moedas.

$P(O \cap Y_1) = P(O|Y_1) * P(Y_1)$

$P(O|Y_1) = P(O_1|S_1) * P(O_2|S_1) = 2/3 * 1/3 = 2/9$

$P(Y_1) = \pi_1 * P(S_1|S_1) = 1/2 * 1/2 = 1/4$

$P(O \cap Y1) = P((O \cap Y_1)|\lambda) = 2/9 * 1/4 \approx 0,055$

- **Para** $Y_2 = [S_1, S_2]$**:**

  $P(O \cap Y_2) = P(O|Y_2) * P(Y_2)$

  $P(O|Y_2) = P(O_1|S_1) * P(O_2|S_2) = 2/3 * 5/6 = 10/18 = 5/9$

  $P(Y_2) = \pi_1 * P(S_2|S_1) = 1/2 * 1/2 = 1/4$

  $P(O \cap Y_2) = P((O \cap Y_2)|\lambda) = 5/9 * 1/4 \approx 0,139$

- **Para** $Y_3 = [S_2, S_1]$**:**

  $P(O \cap Y3) = P(O|Y_3) * P(Y_3)$

  $P(O|Y_3) = P(O_1|S_2) * P(O_2|S_1) = 1/6 * 1/3 = 1/18$

  $P(Y_3) = \pi_2 * P(S_1|S_2) = 1/2 * 1/2 = 1/4$

  $P(O \cap Y_3) = P((O \cap Y_3)|\lambda) = 1/18 * 1/4 \approx 0,014$

- **Para** $Y4 = [S_2, S_2]$**:**

  $P(O \cap Y_4) = P(O|Y_4) * P(Y_4)$

$$P(O|Y_4) = P(O_1|S_2) * P(O_2|S_2) = 1/6 * 5/6 = 5/36$$
$$P(Y_4) = \pi_2 * P(S_2|S_2) = 1/2 * 1/2 = 1/4$$
$$P(O \cap Y_4) = P((O \cap Y_4)|\lambda) = 5/36 * 1/4 \approx 0,035$$

Portanto, a probabilidade de ocorrência da sequência de observações $O = [O_1, O_2]$ poderia ser dada pela soma destas quatro probabilidades, ou seja, de aproximadamente $0,243$.

Podemos também estar interessados em identificar a sequência de estados ocultos que apresenta a maior probabilidade de ter gerado uma determinada observação ("Problema de Decodificação"). No caso do exemplo, a sequência de estados ocultos que maximiza a probabilidade de o modelo ter emitido a sequência de observações $O = [O_1, O_2]$ é a $Y_2 = [S_1, S_2]$.

O exemplo apresentado consiste em um caso onde a probabilidade de emissão de símbolos é discreta, ou seja, temos um número de finito de observações possíveis para as quais são associados valores de probabilidades de emissão de símbolo em função do estado do HMM. A Figura 7 representa graficamente a função massa de probabilidade para este exemplo (vetor de observação discreto).

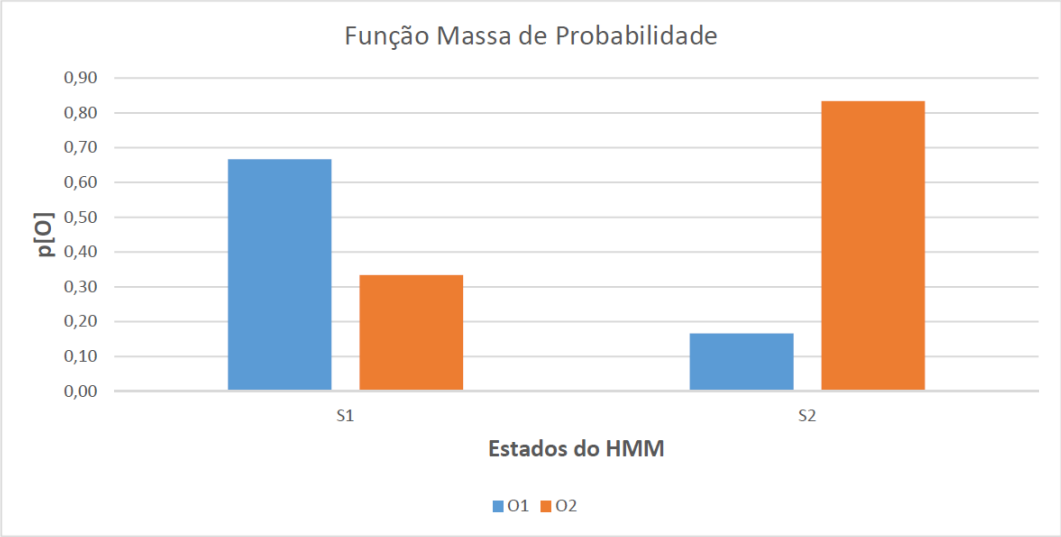


Figura 7 – Distribuição da Probabilidade - Caso Discreto (Lançamento de 2 Moedas).

Entretanto, o caso discreto não consegue representar muitos dos fenômenos que se deseja modelar, como é o caso da voz, devida a sua natureza contínua. Nesses casos utiliza-se f.d.p. contínuas para modelar as probabilidades de emissão de símbolos associada a cada estado do modelo, como é o caso da mistura de gaussianas (GMM), que é emprega neste trabalho.

Podemos considerar um caso onde o vetor de observações possui infinitos valores dentro de uma determinada faixa delimitada pelos limites superior e inferior, que para um

caso unidimensional a mistura de gaussianas (GMM) pode ser demonstrada através da Figura 8, onde são utilizadas 3 gaussianas para modelar a f.d.p. associada a cada estado do modelo. A ideia principal neste caso é encontrar as médias e variâncias de cada uma das gaussianas que, conjuntamente, melhor represente os dados observados.

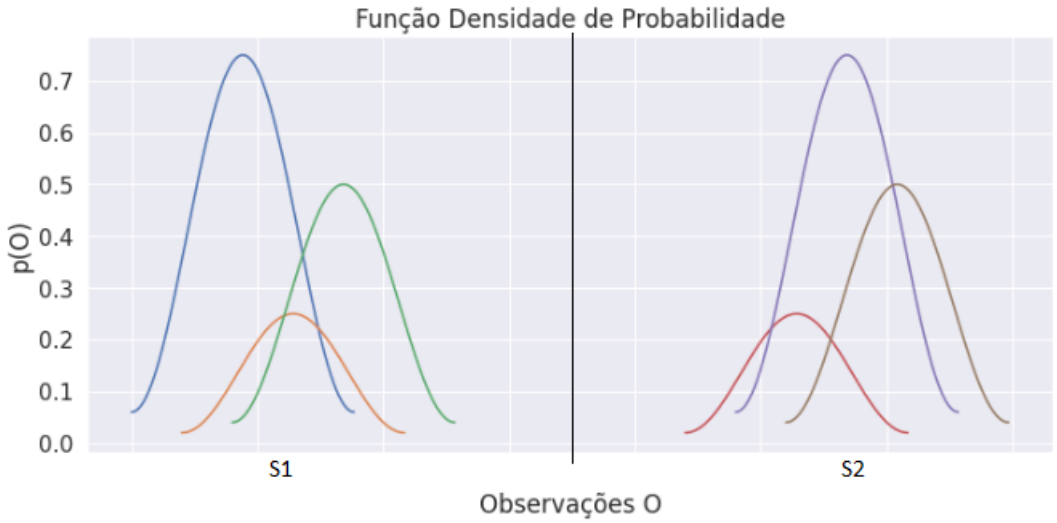


Figura 8 – Distribuição da Probabilidade - Caso Contínuo.

Contudo, cabe ressaltar que este trabalho não tem o objetivo de implementar os algoritmos específicos utilizados para a solução de cada um dos problemas descritos relacionados à implementação do HMM-GMM, tampouco, entrar no mérito dos detalhes de funcionamento desses algoritmos. A ideia aqui é utilizar bibliotecas e pacotes de código consolidados, que nos entregam estes algoritmos prontos para uso no desenvolvimento da aplicação proposta neste trabalho.

2.6 Métricas de Desempenho: Matriz de Confusão, Acurácia, Recall, Precision e F1-score

A Matriz de Confusão é uma ferramenta que pode ser utilizada para a representação gráfica das métricas de desempenho de um modelo de classificação, permitindo avaliar o desempenho do modelo através da comparação dos resultados preditos pelo modelo com os dados reais de entrada do sistema (resultado esperado) (GUANGA, 2018).

Em outras palavras, no contexto da aplicação proposta, queremos verificar se um determinado locutor "A" é de fato reconhecido como locutor "A" dentre os locutores cadastrados na aplicação, bem como queremos avaliar a capacidade do modelo de classificação em identificar um locutor impostor, ou seja, um locutor que não está cadastrado na aplicação.

A Figura 9 apresenta o modelo mais básico de uma Matriz de Confusão, que é o modelo onde temos apenas duas classes possíveis. Este modelo, embora elementar, nos

permite entender a Matriz de Confusão e extrapolar este entendimento para modelos mais complexos, como o que será empregado neste trabalho.

	Preditos: 0	**Preditos: 1**
Esperado: 0	VN	FP
Esperado: 1	FN	VP
Legenda: VN: Verdadeiro Negativo; VP: Verdadeiro Positivo; FP: Falso Positivo; e FN: Falso Negativo		

Figura 9 – Matriz de Confusão Binária. Adaptado de (GUANGA, 2018)

No modelo apresentado, as linhas representam os valores esperados e as colunas representam os valores preditos. As interseções entre linhas e colunas que formam a diagonal principal da matriz (VN e VP) correspondem aos acertos do modelo de classificação e, por sua vez, as interseções entre linhas e colunas que formam a diagonal secundária da matriz (FP e FN) correspondem aos erros do modelo de classificação.

Além da representação gráfica, a partir da Matriz de Confusão é possível extrair uma série de informações e métricas, que nos permitirá avaliar de maneira detalhada e objetiva o desempenho do modelo de classificação, bem como compará-los com estudos de mesma natureza já realizados.

As informações disponibilizadas por uma Matriz de Confusão são as seguintes: verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos (GUANGA, 2018). Essas informações encontram-se detalhadas na subseção 2.6.1.

A partir dessas informações podemos extrair algumas métricas para uma análise mais detalhada do modelo de classificação (GUANGA, 2018). Neste trabalho serão utilizadas as seguintes: Acurácia, *Recall* (sensibilidade), *Precision* e *F1-score*. Essas métricas encontram-se detalhadas na subseção 2.6.2.

2.6.1 Informações

As principais informações extraídas da Matriz de Confusão, contextualizadas para a aplicação proposta neste trabalho, são as seguintes:

- **Verdadeiros Positivos (VP)**: corresponde ao conjunto de locutores que foram reconhecidos adequadamente pela aplicação (i.e., locutor “A” foi reconhecido como locutor “A”, ou seja, aplicação acertou no reconhecimento);
- **Verdadeiros Negativos (VN)**: corresponde ao conjunto de locutores que, de fato, não estão cadastrados na aplicação e que foram reconhecidos como não cadastra-

dos pela aplicação (i.e., identificado um impostor, ou seja, aplicação acertou no reconhecimento – rechaçou impostor);

- **Falsos Positivos (FP)**: corresponde ao conjunto de locutores que, de fato, não estão cadastrados na aplicação e que foram reconhecidos como locutores cadastrados pela aplicação (i.e., aplicação falhou no reconhecimento – aceitou impostor);
- **Falso Negativo (FN)**: corresponde ao conjunto de locutores que, de fato, estão cadastrados na aplicação e que foram reconhecidos como não cadastrados pela aplicação (i.e., aplicação falhou no reconhecimento – rechaçou locutor cadastrado indevidamente).

2.6.2 Métricas

As métricas obtidas a partir das informações detalhadas na subseção 2.6.1, contextualizadas para a aplicação proposta neste trabalho, são as seguintes:

- **Acurácia (ACU)**: corresponde à média global de acerto do classificador ao classificar as classes de locutores (i.e., locutores cadastrados e locutores não cadastrados). Esta métrica pode ser calculada através da Equação 2.8:

$$ACU = \frac{VN + VP}{VP + FP + VN + FN} \tag{2.8}$$

- ***Recall***: também denominado como sensibilidade, corresponde à proporção dos classificados pela aplicação como locutores cadastrados em relação ao total de locutores, de fato, cadastrados na aplicação. Esta métrica pode ser calculada através da Equação 2.9:

$$Recall = \frac{VP}{VP + FN} \tag{2.9}$$

- ***Precision***: também denominado como Valor Predito Positivo (VPP), indica a probabilidade de que um locutor classificado como cadastrado pela aplicação esteja, de fato, cadastrado na aplicação. Esta métrica pode ser calculada através da Equação 2.10:

$$Precision = \frac{VP}{VP + FP} \tag{2.10}$$

- ***F1-Score***: consiste em uma média ponderada obtida a partir da combinação entre *Precision* e *Recall*, sendo indicada para utilização em dados cuja distribuição das classes é desequilibrada. Esta métrica pode ser calculada através da Equação 2.11:

$$F1 - score = 2\frac{Precision * Recall}{Precision + Recall} \tag{2.11}$$

3 Metodologia

O desenvolvimento deste trabalho foi dividido em seis etapas, a fim de obter um caminho lógico para alcançar o objetivo traçado. São elas: i) definição e delimitação da estrutura da aplicação de reconhecimento automático de locutor; ii) pré-processamento do banco de dados; iii) definição das ferramentas computacionais; iv) análise exploratória dos dados e funcionalidades das bibliotecas; v) implementação da aplicação de reconhecimento automático de locutor; e vi) análise de desempenho da aplicação.

3.1 Definição e Delimitação da Estrutura da Aplicação de Reconhecimento Automático de Locutor

A primeira etapa do desenvolvimento foi a definição e delimitação da estrutura funcional da aplicação. Portanto, a fim de visualizar todas as fases de processamento idealizadas para a aplicação proposta, de forma lógica e estruturada, foi elaborado um fluxograma, conforme apresentado na Figura 10.

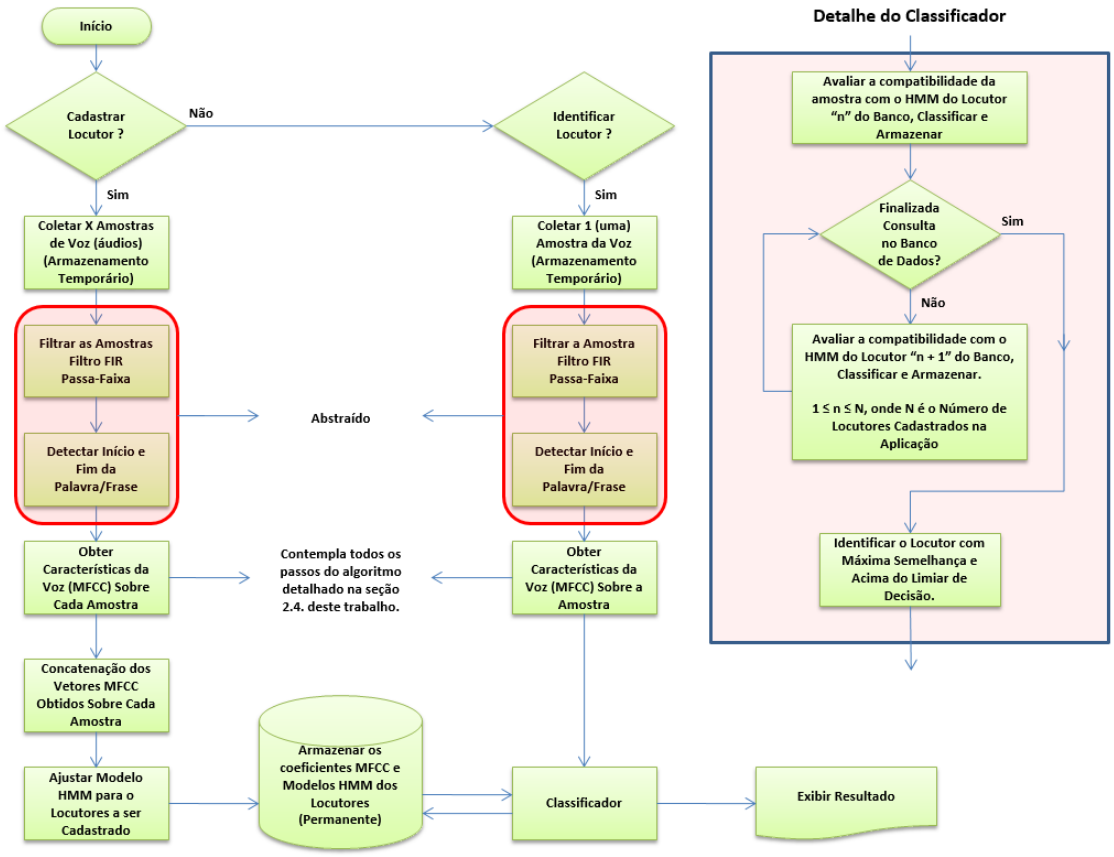


Figura 10 – Fluxograma da Aplicação Proposta.

O fluxograma apresentado na Figura 10 abrange todas as etapas de processamento necessárias para a implementação prática de uma aplicação de reconhecimento automático de locutor. No caso da proposta deste trabalho, *a priori*, iremos abstrair duas das etapas de processamento, visto que trabalharemos com áudios previamente coletados e parametrizados. As duas etapas de processamento que serão abstraídas serão: "Filtrar as Amostras - Filtro FIR - Passa-Faixa" e "Detectar Início e Fim da Palavra/Frase".

As demais etapas serão implementadas e consistem, basicamente, em extrair as características de voz a partir de "X" arquivos de áudio de cada locutor, ou seja, extrair os coeficientes Mel-Cepstrais para cada um desses arquivos de áudio e concatená-los, obtendo uma sequência de vetores de coeficientes Mel-Cepstrais para cada um dos respectivos locutores. Na sequência é estimado um modelo HMM, com f.d.p. de emissões de símbolos modelada por misturas de gaussianas, para cada um desses locutores, os quais são cadastrados no banco de dados da aplicação. Na prática são os coeficientes MFCC e os modelos HMM que ficam armazenados, e não os arquivos de áudio.

A partir desses modelos HMM, na etapa de reconhecimento, é possível avaliar a probabilidade com que cada um dos modelos emite uma determinada sequência de vetores MFCC (sequência de observação), possibilitando a realização do reconhecimento automático do locutor, com base em critérios estabelecidos pelo classificador (máxima verossimilhança e acima do limiar de decisão para rechaçar impostor), quando solicitado.

3.2 Pré-processamento do Banco de Dados

Uma vez definida a estrutura da aplicação, a segunda etapa do desenvolvimento foi realizar o pré-processamento dos dados disponibilizados pela base de dados CEFALA-1, a fim de obter uma estrutura com as características idealizadas, visto que os arquivos de áudio originais disponibilizados são grandes demais, girando em torno de 5 minutos de duração, ou seja, fugindo ao propósito de utilização deste trabalho. Esta etapa foi relativamente simples, visto que os arquivos de áudio disponibilizados pelo CEFALA-1 são acompanhados de um arquivo com extensão *.TextGrid* do software *Praat* (Software aplicado para análise e síntese da fala desenvolvido na Universidade de Amsterdã, disponível para download em: https://www.fon.hum.uva.nl/praat/), com as marcações de segmentação do áudio para cada uma das etapas de coleta.

No caso, o tratamento dos dados consistiu apenas em escolher os trechos de fala de interesse e extrair do áudio original, utilizando o mesmo "Software" mencionado, obtendo áudios com tempo de duração entre 1 e 10 segundos.

Portanto, o banco de dados foi estruturado com 15 locutores, sendo 10 do sexo masculino e 5 do sexo feminino, onde para cada um deles foram selecionados 20 arquivos de áudio, onde cada arquivo corresponde a um conteúdo de fala distinto. Dentre os 20

arquivos de áudio de cada um dos locutores, 16 foram utilizados para o treinamento dos modelos HMM e 4 foram utilizados na fase testes de desempenho da aplicação.

Cabe ressaltar que dos 15 locutores que compõem o banco de dados, apenas 10 foram cadastrados na aplicação, ou seja, são locutores para os quais foram ajustados modelos HMM. Os outros 5 locutores são impostores que foram utilizados para avaliar a capacidade da aplicação em reconhecer e rechaçar impostores.

Seguindo a convenção do protocolo de coleta de dados empregado na formulação da referida Base de Dados (NETO; SILVA; YEHIA, 2019), a relação do conteúdo de fala dos áudios utilizados é a seguinte:

- F01 - “Olha lá o avião azul.”;
- F02 - “Minha mãe namorou um anjo.”;
- F03 – “Sônia, sabe sambar sozinha.”;
- F04 - “Érica tomou suco de pêra e amora.”;
- F05 - “Eu precisei de microfone na conferência.”;
- F06 - “Podia dizer as horas, por favor?”;
- F07 - “A fila aumentou, ao longo do dia.”;
- F08 - “A proposta foi inspecionada pela gerência.”;
- F09 - “Minhas correspondências não estão em casa.”;
- F10 - “As queimadas devem diminuir, este ano.”;
- F11 - “De dia apague a luz.”;
- F12 - “O atabaque do Tito é coberto com pele de gato.”;
- F13 – “Procurei Maria na copa.”;
- F14 - “Daqui a pouco a gente irá pousar.”;
- F15 - “Preciso de ir renovar minha habilitação.”;
- F16 - “Você está jóia?”;
- F17 - “Yuri viu um pequeno jabuti xereta e dez cegonhas felizes comendo kiwi.”;
- F18 - “A rápida raposa marrom pulou sobre o cão preguiçoso.”;
- F19 - “Quem rouba é ladrão, quem rouba muito é barão, quem rouba muito mas esconde passa de barão a visconde.”;
- F20 - “Quando se joga o jogo dos tronos, você vence ou você morre.”.

3.3 Ferramentas Computacionais

Para implementação da aplicação foi utilizada a linguagem de programação Python e o ambiente de programação Jupyter Notebook. A linguagem foi escolhida pela facilidade de utilização e vasta quantidade de bibliotecas e documentação disponíveis, facilmente encontradas na Internet. Já o ambiente de programação por ser gratuito (disponível para download em: https://jupyter.org/) e por sua instalação ser acompanhada das principais bibliotecas da linguagem de programação adotada, o que confere simplicidade e produtividade ao desenvolvimento do trabalho.

Além das bibliotecas comumente utilizadas em códigos desenvolvidos em Python, foram utilizadas bibliotecas específicas para o tipo de aplicação proposta neste trabalho. Nesse contexto, destaca-se a biblioteca *python-speech-features*, a qual é especializada em análise e extração de recursos de voz, implementando o algoritmo da técnica MFCC (LYONS, 2013), bem como a biblioteca *hmmlearn*, a qual é especializada em algoritmos de aprendizagem supervisionada e inferências de HMM (HMMLEARN, 2021), sendo esta última empregada para a implementação do classificador de reconhecimento automático de locutor que compõe a aplicação objeto de estudo deste trabalho.

Por fim, outra biblioteca utilizada que merece destaque especial foi a *scikit-learn*, especificamente o pacote *sklearn.metrics* (disponível para download em: https://scikit-learn.org/stable/), que disponibiliza uma função para a geração de matrizes de confusão, utilizadas neste trabalho para avaliação de desempenho da aplicação.

3.4 Análise Exploratória dos Dados e das Funcionalidades das Bibliotecas

Em relação à análise exploratória realizada sobre os dados disponibilizados pela base de dados CEFALA-1, através do software *Praat*, foram feitos os testes de extração dos trechos de fala de interesse, conforme tratado na seção 3.2 que versa sobre o pré-processamento do banco de dados. A Figura 11 apresenta a interface do software *Praat* que permite visualizar a segmentação do áudio, desenvolvida e disponibilizada pela base de dados CEFALA-1 junto com o arquivo de áudio. No caso o trecho do áudio de interesse está destacado na figura, correspondendo ao segmento F01, cujo conteúdo da fala é: “Olha lá o avião azul”. Ao realizar a extração do trecho de interesse é gerado um novo arquivo de áudio, de mesmo formato do original, no caso WAV, mas contendo somente o trecho selecionado.

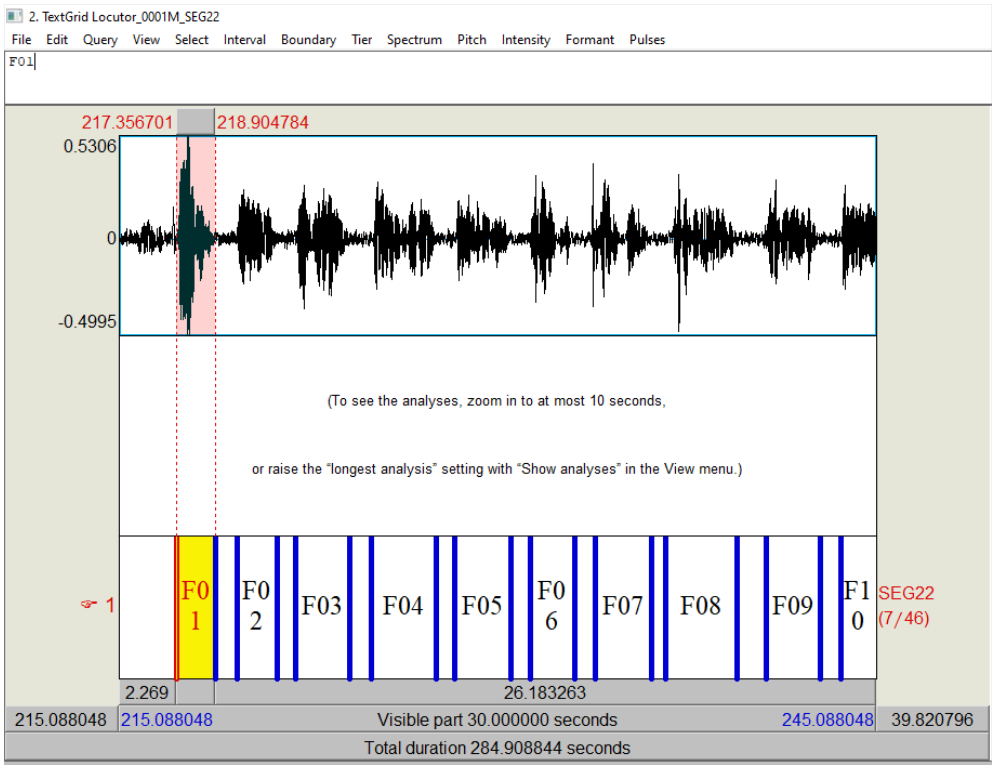


Figura 11 – Extração do Trecho de Áudio de Interesse.

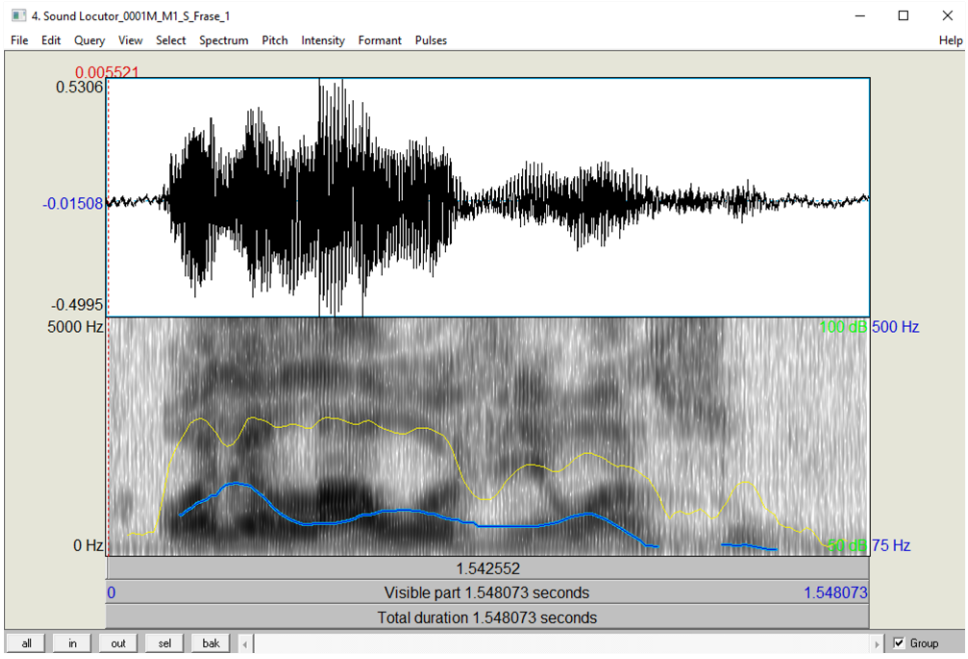


Figura 12 – Detalhe do Trecho de Áudio de Interesse.

A Figura 12 apresenta em detalhes o trecho de áudio extraído na etapa anterior e o seu respectivo espectrograma, ambos obtidos através do software *Praat*, o qual permite realizar análises espectrais sobre o arquivo de áudio e, no contexto desta análise, servem ao propósito de parâmetro de comparação para nortear as etapas de desenvolvimento da aplicação de identificação automática de locutor proposta neste trabalho.

Em relação à análise exploratória do arquivo de áudio, realizada através da biblioteca *python-speech-features*, foi implementada a parte inicial do código da aplicação, onde o objetivo foi testar a importação e a reprodução do arquivo de áudio, a obtenção dos coeficientes MFCC e a representação gráfica do sinal de áudio através do ambiente de programação. O código implementado cumpriu com o planejado para esta etapa. A Figura 13 apresenta o resultado obtido para a representação gráfica do sinal de áudio no domínio do tempo, da frequência, do tempo-frequência (espectrograma) e os correspondentes coeficientes MFCC.

Cabe ressaltar que o sinal áudio analisado nesta etapa foi aquele extraído durante a realização da análise exploratória através do software *Praat*. As Figuras 14 e 15 apresentam os dados numéricos das principais características obtidas nesta análise, a saber: tamanho do sinal, tempo de duração do sinal, frequência de amostragem, período de amostragem e coeficientes MFCC.

Traçando um paralelo do algoritmo detalhado na seção 2.4 do referencial teórico com os coeficientes MFCC apresentados na Figura 15, obtidos através da biblioteca *python-speech-features*, evidenciamos a correspondência do resultado prático com o passo-a-passo do referido algoritmo, ou seja, para cada segmento do áudio oriundo do processo de janelamento do sinal obtivemos um vetor de 13 coeficientes MFCC. A descrição do algoritmo faz referência ao número de 26 coeficientes, mas a função implementada na biblioteca *python-speech-features*, por padrão, retorna somente os 13 primeiros, os quais atendem ao propósito da aplicação objeto de estudo.

A função *python-speech-features.base.mfcc* da biblioteca *python-speech-features* faz a segmentação do sinal de áudio em janelas de 25 ms (valor padrão), com cada janela tomada a cada 10 ms. Portanto, no caso do áudio analisado, cuja duração é de aproximadamente 1,54 s, o mesmo foi dividido em 154 janelas e, consequentemente, resultou em uma sequência de 154 vetores de coeficientes MFCC, onde cada vetor possui 13 coeficientes, conforme apresentado na Figura 15.

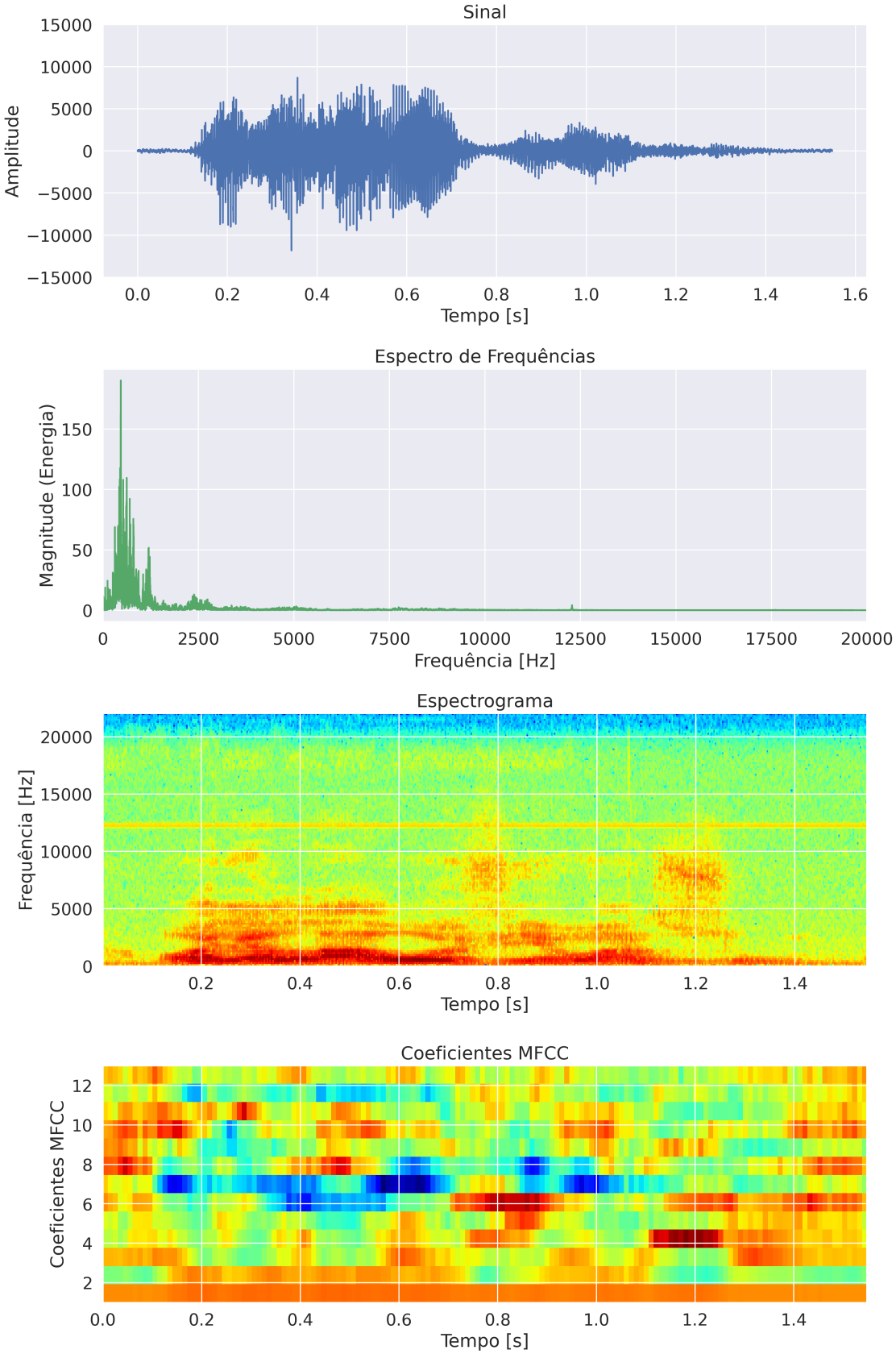


Figura 13 – Representação Gráfica do Sinal de Áudio no Domínio do Tempo, da Frequência, Tempo-Frequência (Espectrograma) e Correspondentes Coeficientes MFCC.

```
[21] # Importando e Reproduzindo um Sinal de Áudio
     # Conteúdo de fala: "Olha lá o avião azul"

     fname = '/content/drive/MyDrive/TrabalhoGraduacao/'+'Locutor_01M_A4_F01.wav'

     rate, data = wavfile.read(fname)

     IPython.display.Audio(fname)
```


```
[22] # Obtenção dos Parâmtros do Sinal de Áudio
     ts = 1/rate
     tw = data.size/rate
     t = np.linspace(0, tw, int(tw/ts))

     print("Tamanho do sinal [amostras]:", data.size)
     print("Tempo de duração do sinal [s]:", tw)
     print("Frequência de Amostragem [Hz]: ", rate)
     print("Período de Amostragem do Sinal - Ts[s]: ", ts)

     Tamanho do sinal [amostras]: 68270
     Tempo de duração do sinal [s]: 1.5480725623582767
     Frequência de Amostragem [Hz]:  44100
     Período de Amostragem do Sinal - Ts[s]:  2.2675736961451248e-05
```

Figura 14 – Parâmetros do Sinal de Áudio.

```
[35] # Imprimindo os Dois Primeiros Vetores de Coeficientes MFCC do Sinal
     print('Os Dois Primeiros Vetores de Coeficientes MFCC do Sinal:\n')
     print(coef_mfcc_plot[0][0:2])

     # Imprimindo a Quantidade de Vetores de Coeficientes MFCC do Sinal
     print('\nQuantidade de Vetores de Coeficientes MFCC do Sinal: ',
           len(coef_mfcc_plot[0]))

     # Imprimindo o Tamanho dos Vetores de Coeficientes MFCC do Sinal
     print('\nO Tamanho dos Vetores de Coeficientes MFCC do Sinal: ',
           len(coef_mfcc_plot[0][0:2][0]))

     Os Dois Primeiros Vetores de Coeficientes MFCC do Sinal:

     [[ 12.48704343 -19.19441957   7.36206364 -10.4570657   -5.02174943
        12.37766557   4.16511198  20.40914654  12.65957656   9.26358489
         2.81915672  -6.75845507   5.22533742]
      [ 12.48202773 -19.19386183   7.10415905 -10.73596796  -7.50244542
         8.10957761  -1.62698738  19.30370477  11.15917205  10.60992094
        -0.66668686  -5.38816894   7.9982328 ]]

     Quantidade de Vetores de Coeficientes MFCC do Sinal:  154

     O Tamanho dos Vetores de Coeficientes MFCC do Sinal:  13
```

Figura 15 – Vetores de Coeficientes MFCC Correspondentes aos Dois Primeiros Quadros Oriundos do Processo de Janelamento do Sinal.

Após a realização da análise exploratória dos dados e implementação do código para obtenção dos coeficientes MFCC, a etapa seguinte foi a implementação do código para o treinamento e avaliação dos modelos HMM-GMM, onde foi treinado um modelo para cada um dos 10 locutores que seriam cadastrados na aplicação. Nesta etapa cada um dos modelos foi treinado sem a utilização do modelo de fundo (UBM, que será descrito na seção 3.5), ou seja, foi treinado a partir das respectivas amostras de áudios de treinamento reservada para cada um dos locutores individualmente. Esses modelos foram concebidos arbitrariamente com 2 estados e 1 gaussiana para modelar a função densidade de probabilidade. O principal objetivo desta etapa foi a realização da análise exploratória da biblioteca *hmmlearn*.

A proporcionalidade entre locutores do sexo masculino e feminino, bem como a proporcionalidade entre a quantidade de amostras para treinamento dos HMMs e testes de reconhecimento foram escolhidas sem nenhum critério técnico, ou seja, foram tomados os 10 primeiros locutores do banco de dados estruturado na etapa de pré-processamento e os 10 primeiros áudios de cada um dos respectivos locutores.

Os áudios utilizados para a etapa de treinamento dos HMMs foram os seguintes:

- F01 - “Olha lá o avião azul.”;
- F02 - “Minha mãe namorou um anjo.”;
- F03 – “Sônia, sabe sambar sozinha.”;
- F04 - “Érica tomou suco de pêra e amora.”;
- F05 - “Eu precisei de microfone na conferência.”;
- F06 - “Podia dizer as horas, por favor?”;

Os áudios utilizados para a etapa de teste de reconhecimento dos locutores foram os seguintes:

- F07 - “A fila aumentou, ao longo do dia.”;
- F08 - “A proposta foi inspecionada pela gerência.”;
- F09 - “Minhas correspondências não estão em casa.”;
- F10 - “As queimadas devem diminuir, este ano.”;

Cabe ressaltar que todos os arquivos de áudio utilizados neste trabalho foram gravados em um mesmo ambiente e utilizando o mesmo microfone (i.e., microfone de um Smartphone Samsung Galaxy S2 Lite GT-i9070, com tecnologia MEMS (*Micro-ElectroMechanical Systems*), o qual é nomeado como M4 no trabalho (NETO, 2018)).

A proporcionalidade de arquivos de áudio utilizados para o treinamento dos HMMs e testes de reconhecimento, foram respectivamente, 0,6 (sessenta por cento) e 0,4 (quarenta por cento). Esta proporcionalidade foi arbitrada e não respeitou a nenhum critério técnico.

Nesta primeira versão do classificador não foi implementada nenhuma regra, como um limiar de verossimilhança para rechaçar impostor (i.e., locutor que tenta ser reconhecido pela aplicação sem que, de fato, esteja cadastrado na mesma). A ideia central para esta análise inicial foi simplesmente saber se o classificador era capaz de reconhecer adequadamente os locutores cadastrados.

Após as considerações e delimitações supracitadas, a Figura 16 apresenta o trecho do código desenvolvido para o classificador, o qual calcula a probabilidade de que cada modelo HMM cadastrado tenha emitido uma dada sequência de vetores de observação MFCC, reportando uma lista com a probabilidade associada a cada modelo e a posição da lista que apresenta a maior probabilidade de ter emitido tal vetor de observação. A posição da lista reportada corresponde à identificação numérica do locutor cadastrado no Banco de Dados da aplicação.

```
[13] def classificador(X, modelos):
       logProb = []
       for model in modelos:
         logProb.append(model.score(X))
       return logProb.index(np.array(logProb).max())+1, logProb
       # Observação: "+1" é necessário porque o vetor de modelos vai de 0 a N-1 locutores.
       # Portanto ao adicionar "+1" reposta-se o número do locutor entre 1 a N.
```

```
[14] # Teste de reconhecimento utilizando as 4 locuções de teste para cadas um dos locutores

     # Vetor com as locuções dos locutores a serem reconhecidos.
     # Nesta etapa todas as locuções neste vetor são de locutores cadastrados.
     # Porém são locuções que não foram utilizadas no treinamento dos modelos.
     X = dic_vetor_observacoes_mfcc_reconhecimento_HMM
     N = 10 # Número de locutores cadastrados
     nL = 4 # Número de locuções de cada um dos locutores para teste

     locutores_reais=[]
     locutores_preditos=[]

     for i in range(1, N+1):
       for j in range(0, nL):
         locutores_reais.append(i)
         locutores_preditos.append(classificador(X[i][j], modelos_HMM)[0])

     #Saídas de teste
     print(locutores_reais)
     print(locutores_preditos)

     [1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 7, 7, 7, 7, 8, 8, 8, 8, 9, 9, 9, 9, 10, 10, 10, 10]
     [1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 7, 7, 9, 7, 8, 8, 8, 8, 9, 9, 9, 9, 10, 10, 10, 10]
```

Figura 16 – Trecho do Código Referente ao Classificador (Primeira Versão).

Analisando o trecho de código apresentado na Figura 16 observa-se que o classificador é empregado na tarefa de identificação de locutores para as 40 amostras de teste, sendo 4 amostras por locutor. Cabe ressaltar que a etapa de treinamento dos modelos HMMs não está retratada neste trecho de código. O resultado numérico preliminar apresentado nesta figura pode ser melhor interpretado no gráfico da Figura 17, bem como na Matriz de Confusão apresentada na Figura 18.

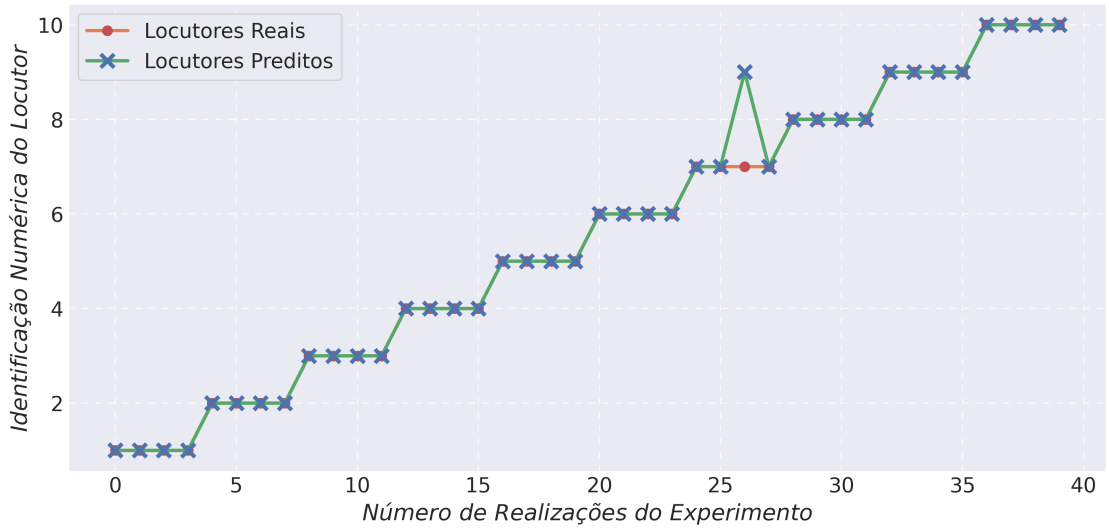


Figura 17 – Resultado Preliminar dos Testes de Reconhecimento.

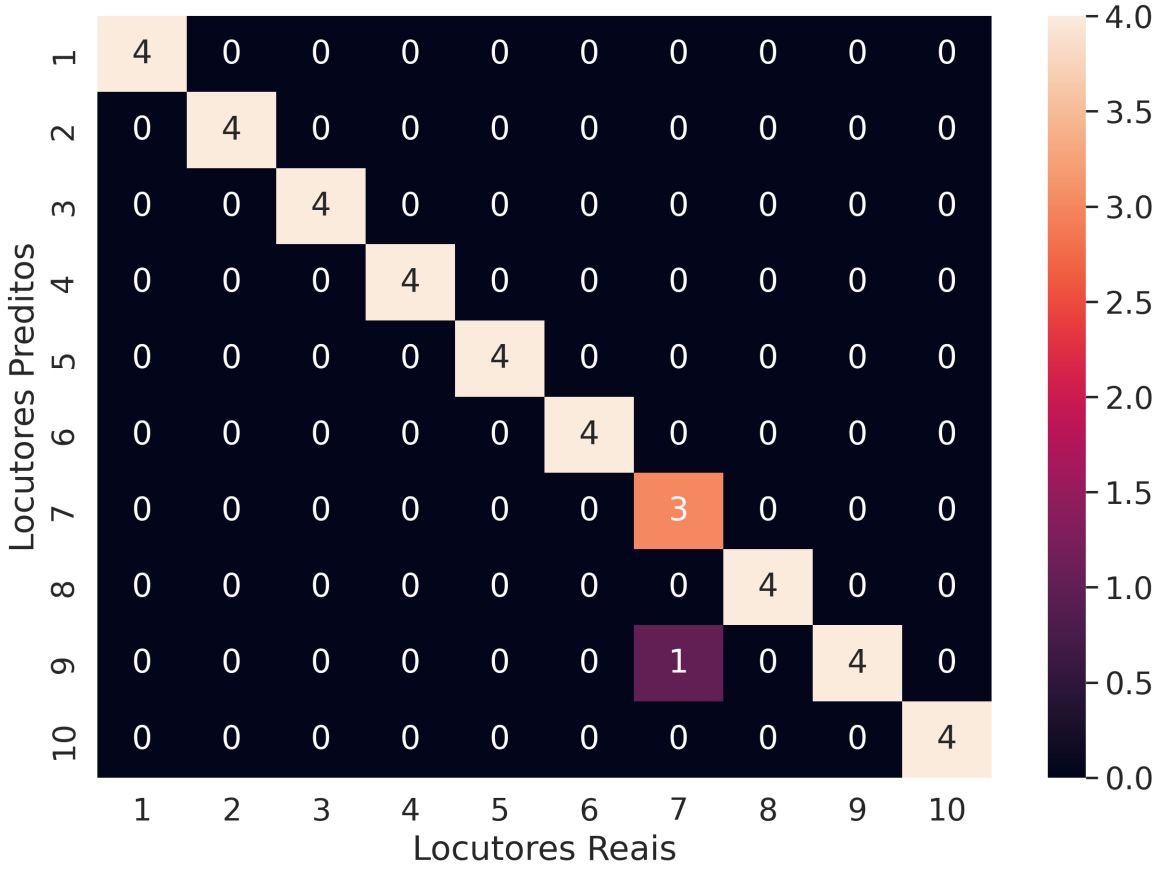


Figura 18 – Matriz de Confusão para o Resultado Preliminar dos Testes de Reconhecimento.

Como pode ser observado, das 40 solicitações de reconhecimento, o classificador errou apenas 1. Com o objetivo de avaliar a consistência e repetibilidade do resultado obtido, o processo de identificação foi realizado repetidas vezes e foi observado que o mesmo erro foi reportado pelo classificador em todas as realizações, ou seja, a amostra de áudio correspondente ao conteúdo de fala "F09 – Minhas correspondências não estão em casa." do locutor n° 7 foi identificada como sendo do locutor n° 9. Em análise qualitativa (i.e., escutando áudios de ambos os locutores), percebe-se que a cadência de fala e o timbre das vozes são parecidos, embora o locutor n° 7 seja do sexo masculino e o locutor n° 9 seja do sexo feminino.

Analisando a funcionalidade da versão inicial da aplicação como um todo, constatou-se que existem alguns parâmetros que podem ser trabalhados com o objetivo de maximizar o seu desempenho, tais como: diminuir o tamanho dos quadros oriundos do processo de janelamento realizado na etapa de extração dos coeficientes MFCC, obtendo assim maior quantidade de vetor a partir de cada uma das amostras de áudio; utilizar mais amostras de áudio para a etapa de treinamento dos HMMs que representam os locutores; utilizar somente 12 coeficientes MFCC (2 ou 13), que é o recomendado pelo referencial teórico (no caso, foi utilizado os 13 primeiros na versão inicial); e testar a utilização de HMMs com diferentes números de estados e diferentes números de gaussianas para modelar a f.d.p. (no caso, foi utilizado 2 estados e 1 gaussiana para modelar a f.d.p.).

A partir da Matriz de Confusão foram extraídos, nesta etapa apenas em caráter de análise exploratória, os valores correspondentes às seguintes métricas de desempenho e análise de resultados, a saber: *Precision*, *Recall*, *F1-score* e Acurácia, conforme apresentado na Tebela 2.

Tabela 2 – Métricas de Desempenho e Análise dos Resultados.

-	Precision	Recall	F1-score	Suporte
1	1	1	1	4
2	1	1	1	4
3	1	1	1	4
4	1	1	1	4
5	1	1	1	4
6	1	1	1	4
7	0.75	1	0.86	3
8	1	1	1	4
9	1	0.80	0.89	5
10	1	1	1	4
Acurácia	-	-	0.97	40
Macro avg	0.97	0.98	0.97	40
Weighted avg	0.98	0.97	0.98	40

No contexto dos testes preliminares, onde todos os locutores testados, de fato, estão cadastrados na aplicação, o valor obtido para a Acurácia, ou seja, para média global de acertos do classificador, foi de 0,97 (noventa e sete por cento).

Contudo, a análise exploratória foi fundamental para relacionar a teoria apresentada no referencial teórico com a aplicação prática das ferramentas computacionais escolhidas para o processamento dos sinais de áudio, extração de coeficientes MFCC e modelagem dos HMM-GMM, a fim de obter modelos probabilísticos capazes de representar as características vocais dos locutores.

3.5 Implementação da Aplicação de Reconhecimento Automático de Locutor

Após as implementações iniciais realizadas a título de análise exploratória, onde foram verificadas as possibilidades de ajustes de parâmetros para maximização do desempenho da aplicação, iniciou-se a etapa de codificação da versão atual, onde algumas dessas possibilidades foram testadas.

Em relação à quantidade de coeficientes MFCC, na análise exploratória foram utilizados os 13 primeiros coeficientes MFCC e na versão atual foram utilizados apenas 12, do 2° ou 13°, conforme embasado no referencial teórico. A Figura 19 apresenta o trecho do código da aplicação correspondente à função desenvolvida para extração dos coeficientes MFCC de uma amostra de áudio. A função recebe como parâmetro o nome do arquivo com extensão .wav, realiza a leitura dos parâmetros deste e realizada a extração dos respectivos coeficientes MFCC, retornando os respectivos vetores com os 12 valores de interesse, ou seja, do 2° ao 13° coeficiente MFCC.

```
[33] def extracao_mfcc(fname):
       rate, data = wavfile.read(fname)
       mfcc_13 = mfcc(data, rate, nfft = 2048, winfunc = np.hamming)
       mfcc_12 = []
       for m in mfcc_13:
         mfcc_12.append(list(np.delete(m,[0])))
       return np.array(mfcc_12)
```

Figura 19 – Trecho de Código para Extração dos Coeficiente MFCC

As amostras de áudio foram separadas em bancos de dados de teste e treinamento, a fim de atender aos critérios estabelecidos para a realização da análise de desempenho da aplicação detalhada na seção 3.6 deste documento. Esses bancos de dados compostos de amostras de áudio com extensão .wav deram origem aos correspondentes compostos pelos coeficientes MFCC.

Na análise exploratória foram utilizadas apenas 6 amostras de áudio para treinamento e 4 amostras de áudio para teste para o modelo HMM de cada um dos locutores. Na versão atual foram utilizadas 16 amostras para treinamento e mantida a quantidade de 4 amostras para testes. Nessas condições, a proporcionalidade de arquivos de áudio

utilizados para o treinamento dos HMMs e testes de reconhecimento passaram de 0,6 (sessenta por cento) e 0,4 (quarenta por cento) para 0,8 (oitenta por cento) e 0,2 (vinte por cento), respectivamente.

Em relação à quantidade de estados para modelar os HMMs e a quantidade de gaussianas para modelar as f.d.p. de emissão de símbolos, na análise exploratória foram adotados 2 estados e 1 gaussiana e na versão atual foram adotados 2 estados e 6 gaussianas.

Os sistemas de verificação de locutor baseados em Modelos Ocultos de Markov são estruturados de maneira que cada locutor é representado por um HMM que é treinado e armazenado na aplicação. Dessa maneira, a tarefa de verificação consiste em calcular a probabilidade de uma amostra de áudio ter sido emitida por um HMM de referência. O locutor deve ser aceito como válido, ou seja, como cadastrado na aplicação, se o valor da probabilidade associado ao respectivo modelo for maior ou igual a um limiar pré-estabelecido e rechaçado caso contrário (FECHINE, 1994). Neste caso, a tarefa de verificação pressupõe que o locutor de alguma maneira informa para a aplicação a sua identificação e a tarefa da aplicação é simplesmente validar se o locutor é, de fato, quem ele diz que é. Embora este conceito não seja exatamente o mesmo proposto para a aplicação objeto de estudo, serviu de referência para a compreensão da estratégia de rechaçar locutores impostores.

No caso da aplicação objeto de estudo a ideia é realizar o reconhecimento automático do locutor, ASR. Portanto, a tarefa de reconhecimento consiste em calcular a probabilidade de que uma amostra de áudio tenha sido emitida por cada um dos modelos HMM previamente cadastrados na aplicação, reportando aquele que apresenta a maior probabilidade de ter emitido a respectiva amostra de áudio. Uma vez identificado o modelo que possui a maior probabilidade de ter emitido tal amostra de áudio, a tarefa passa a ser de verificação, ou seja, verificar se de fato o locutor reconhecido pela aplicação é um locutor válido ou não, com base em um limiar pré-estabelecido. Um locutor reportado como não válido corresponde a um locutor impostor.

A técnica de reconhecimento de padrões empregada neste trabalho para o reconhecimento automático de locutores foi o HMM-GMM descrito no referencial teórico, onde foram detalhados os parâmetros que envolvem o HMM propriamente dito. A descrição acerca da mistura de gaussianas (GMM) que modela a f.d.p. de emissão de símbolos será tratada nesta seção, a fim de relacioná-la com a técnica de obtenção do GMM, que incorpora um modelo de fundo, chamado de UBM, dando origem ao UBM-GMM, que no contexto deste trabalho consiste em uma tentativa de reprodução do modelo detalhado no trabalho (NETO, 2018).

A modelagem das f.d.p. realizadas através da mistura de gaussianas (GMM) para cada um dos locutores, a partir de um único modelo de fundo (UBM), nos permite a implementação completa e eficiente de uma solução de reconhecimento automática de

locutor, conforme idealizado para este trabalho.

O UBM é um modelo global treinado via EM com base em todas as amostras de áudio disponíveis na base de dados de treinamento, ou seja, com base em todas as amostras de áudio que foram utilizadas para o cadastro de todos os locutores na aplicação. Nesta concepção, os modelos GMMs que representam cada um dos locutores individualmente são obtidos a partir do UBM, através do processo de adaptação chamado MAP, o qual realiza a adaptação dos parâmetros da mistura de gaussianas obtida no treinamento do UBM. Essa adaptação é feita através do retreinamento do UBM utilizando novas amostras de áudio do locutor para o qual se deseja obter um GMM (no caso deste trabalho foram utilizadas as mesmas amostras de áudio individuais para o locutor a ser cadastrado na aplicação) (NETO, 2018). Este processo de adaptação é representado na Figura 20.

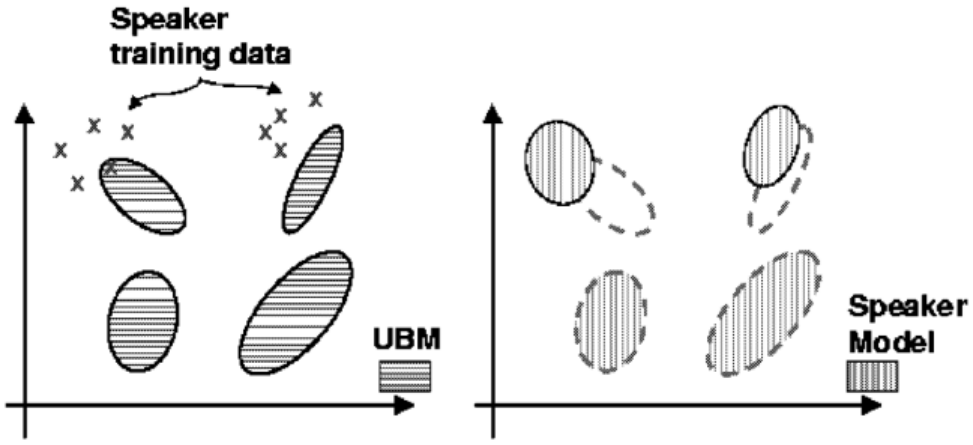


Figura 20 – Processo de Obtenção de um GMM a partir de um UBM. (REYNOLDS; QUATIERI; DUNN, 2000 apud NETO, 2018).

Uma vez obtidos os respectivos bancos de dados de coeficientes MFCC de treinamento e definida a estratégia de treinamento dos HMMs foram implementadas as funções para obtenção do modelo de fundo UBM e dos GMMs individuais para cada um dos locutores a serem cadastrados na aplicação. As Figuras 21 e 22, apresentam os trechos do código da aplicação correspondentes às funções desenvolvidas para obtenção do UBM e dos GMMs, respectivamente.

A função para obtenção do UBM recebe um dicionário do Python com todos os coeficientes MFCC de treinamento, concatena esses coeficientes, treina e retorna o modelo UBM obtido. O modelo UBM retornado carrega todos os atributos de um HMM-GMM, quais sejam: matriz de probabilidade inicial de estados, matriz de probabilidade de transição entre estados e matrizes de pesos, médias e covariâncias das misturas de gaussianas.

A função para obtenção dos GMMs recebe um dicionário do Python com todos os coeficientes MFCC de treinamento e o modelo UBM. A cada iteração do laço “for”, os

```
[36] # Função para Obtenção do modelo de fundo UBM
     def UBM(dic_vetor_observacoes_mfcc_treinamento_HMM):
       Y = dic_vetor_observacoes_mfcc_treinamento_HMM
       vetor_mfcc_treinamento_HMM_global = []
       for i in Y:
         for j in Y[i]:
           vetor_mfcc_treinamento_HMM_global.extend(j)
       modelo_UBM = hmm.GMMHMM(n_components = 2, n_mix=6, covariance_type = 'diag')
       modelo_UBM.fit(vetor_mfcc_treinamento_HMM_global)

       print(modelo_UBM.startprob_)
       print(modelo_UBM.transmat_)

       return modelo_UBM
```

Figura 21 – Trecho de Código para Obtenção do UBM.

coeficientes MFCC de um locutor são concatenados de maneira a compor um vetor de coeficientes para este locutor, o qual é utilizado para realizar o treinamento de um GMM para este locutor, através da adaptação MAP dos parâmetros obtidos do modelo UBM (matrizes de pesos, médias e covariâncias). A função reporta um vetor de modelos GMM, sendo um para cada locutor cadastrado.

```
[37] # Função para Obtenção dos Modelos Individuais (GMMs) para cada um dos Locutores
     # Função retorna um vetor de modelos GMM, sendo um para cada locutor cadastrado
     def GMM(dic_vetor_observacoes_mfcc_treinamento_HMM, modelo_UBM):
       Y = dic_vetor_observacoes_mfcc_treinamento_HMM
       vetor_modelos_GMM=[]

       UBM_local = modelo_UBM
       weights = UBM_local.weights_
       means = UBM_local.means_
       covars = UBM_local.covars_

       for i in Y:
         vetor_mfcc_treinamento_HMM = []

         model = hmm.GMMHMM(n_components = 2, n_mix=6, covariance_type = 'diag',
                            algorithm='map', params='mcw', init_params='st')
         model.weights_ = weights
         model.means_ = means
         model.covars_ = covars

         for j in range(len(Y[i])):
           vetor_mfcc_treinamento_HMM.extend(Y[i][j])
         model.fit(vetor_mfcc_treinamento_HMM)
         vetor_modelos_GMM.append(model)
       return vetor_modelos_GMM
```

Figura 22 – Trecho de Código para Obtenção dos GMMs dos Locutores.

A Figura 23 apresenta uma estrutura simplificada de um classificador que utiliza a estratégia UBM-GMM. A funcionalidade deste classificador consiste, basicamente, em

confrontar um vetor de coeficientes MFCC, que na figura está representado pelo bloco *"Front-end Processing"*, com os modelos UBM e GMM obtidos previamente, representados pelos blocos *"Background model"* e *"Hyp. Speaker model"*, respectivamente. No caso da aplicação objeto de estudo, o modelo GMM é o de maior probabilidade de ter emitido o vetor de coeficientes MFCC, dentre os modelos cadastrados. A saída de cada um desses blocos consiste no valor do logaritmo da probabilidade deles terem emitido este vetor de coeficientes, sobre as quais é realizada a operação "Somatória", cuja saída deve ser comparada com um limiar a ser ajustado previamente. Caso o valor obtido seja maior que o limiar, o locutor deve ser considerado como pertencente ao locutor representado pelo GMM; caso contrário, deve ser rechaçado, pois a probabilidade de ele ter sido emitido pelo GMM é menor do que ter sido emitido pelo UBM, que, via de regra, representa o impostor, ou seja, uma alternativa ao locutor representado pelo GMM (NETO, 2018).

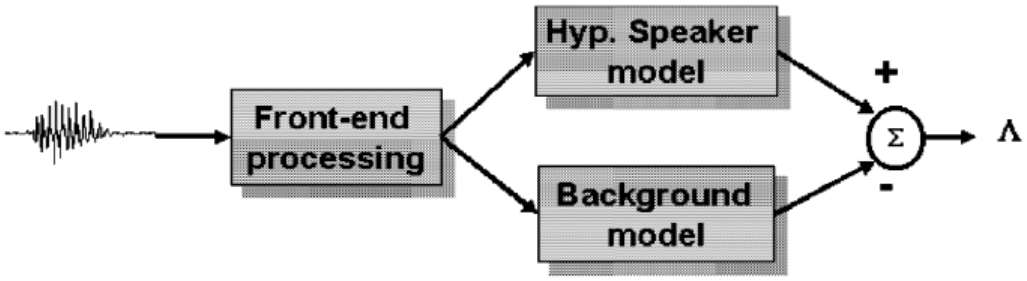


Figura 23 – Sistema de Verificação de Locutor Baseado no Teste de Verossimilhança. (REYNOLDS; QUATIERI; DUNN, 2000 apud NETO, 2018).

A modelagem para este classificador consiste na realização do seguinte teste de hipótese (NETO, 2018):

I) Dado um vetor de coeficientes MFCC de uma amostra de áudio X de um locutor hipotético S, a tarefa de verificar se este vetor foi, de fato, emitido pelo locutor consiste em testar as seguintes hipóteses:

$$H_0 : X \textit{ pertence ao locutor } S; \quad H_1 : X \textit{ não pertence ao locutor } S. \tag{3.1}$$

II) A decisão entre as hipóteses H_0 e H_1 é realizada através do teste de verossimilhança apresentado na Equação 3.2.

$$\Lambda(X) = \frac{p(X|H_0)}{p(X|H_1)} : Se\ \Lambda(X) \geq \theta,\ Aceita\ H_0; \quad Se\ \Lambda(X) < \theta,\ Rejeita\ H_0. \tag{3.2}$$

III) Entretanto, dado que o modelo GMM representa o locutor cadastrado com maior probabilidade de ter emitido o vetor de coeficientes MFCC X e o UBM representa

uma alternativa a este locutor (que pode ser considerado um impostor), a razão de verossimilhança para a amostra X pode ser generalizada pela Equação 3.3:

$$\Lambda(X) = \frac{p(X|GMM)}{p(X|UBM)} = \log p(X|GMM) - \log p(X|UBM) \tag{3.3}$$

IV) Caso $\Lambda(X) \geq \theta$, vetor de coeficientes MFCC é classificado como pertencente ao modelo GMM (locutor cadastrado que apresenta maior probabilidade de ter emitido o vetor). Caso contrário, o vetor é classificado como pertencente ao UBM (impostor).

A Figura 24 apresenta o trecho de código da aplicação que corresponde à função desenvolvida para o classificador.

```
[88] def classificador(X, vetor_modelos_GMM, modelo_UBM):
       logProb = []
       for model in vetor_modelos_GMM:
         logProb.append(model.score(X))

       maxProbIndividual = np.array(logProb).max()
       probGlobal = modelo_UBM.score(X)

       # Regra para rechaçar o impostor
       if maxProbIndividual - probGlobal <= 0:
         return 0,  probGlobal, maxProbIndividual
       else:
         return logProb.index(np.array(logProb).max())+1, probGlobal, maxProbIndividual
         # Observação: "+1" é necessário porque o vetor de modelos vai de 0 a N-1 locutores.
         # Portanto ao adicionar "+1" reporta-se o número do locutor entre 1 a N.
```

Figura 24 – Trecho de Código da Função do Classificador.

A função do classificador recebe um vetor de coeficientes MFCC correspondente a uma amostra de áudio de um locutor qualquer, o vetor de modelos GMMs contendo um modelo GMM para cada locutor cadastrado e o modelo UBM. A função identifica o modelo GMM de maior probabilidade de ter emitido o vetor de coeficientes MFCC e realiza o teste de hipótese descrito anteriormente. Retornando, dentre outras informações, o valor 0 caso seja um locutor impostor e o número do locutor reconhecido caso seja um locutor válido, ou seja, de 1 a N, onde N é o número de locutores cadastrados na aplicação.

3.6 Análise de Desempenho da Aplicação

A validação cruzada é uma prática comum para avaliar modelos de reconhecimento de padrões de aprendizado supervisionado, que consiste, basicamente, em realizar sucessivas divisões do banco de dados em dois bancos menores, um para treinamento e outro para teste do modelo (SCIKITLEARN, 2022).

Para o teste da aplicação desenvolvida, cada locutor foi treinado com 16 amostras de áudio e testado com outras 4 amostras de áudio. Aplicando a validação cruzada, conforme apresentado na Figura 25 obtivemos 5 *splits* (divisões do banco de dados).

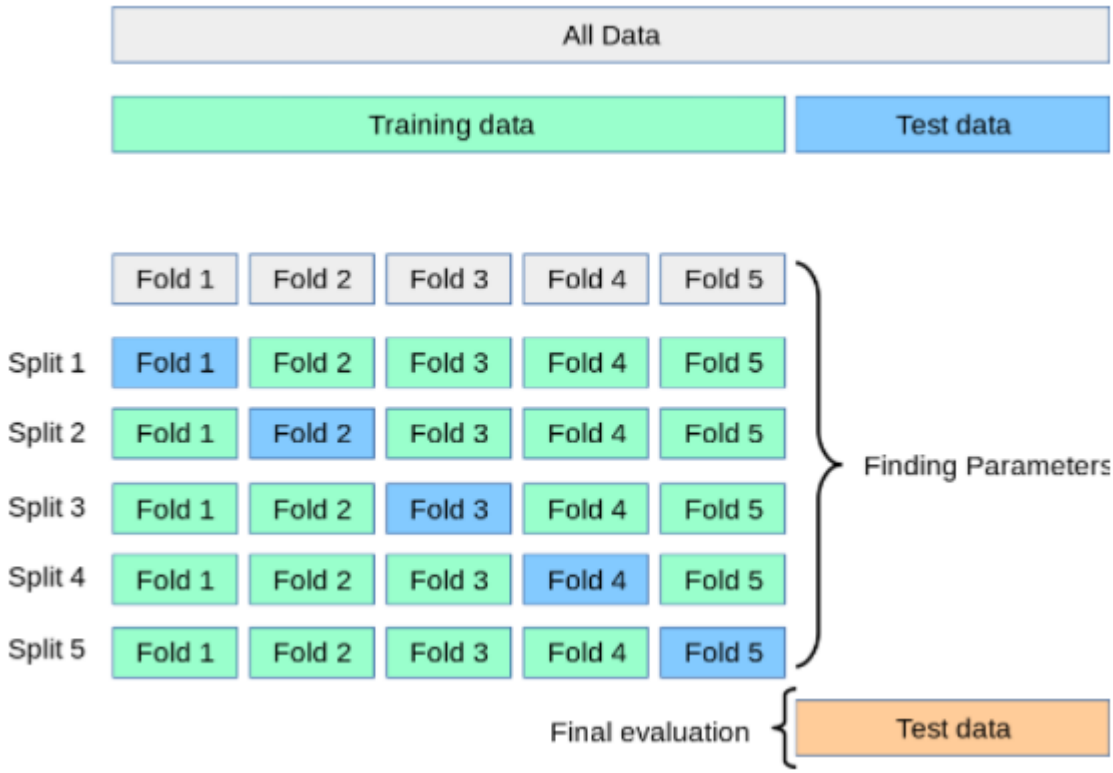


Figura 25 – Esquema de Validação Cruzada (SCIKITLEARN, 2022).

Analisando a Figura 25 nota-se a presença de um terceiro banco de dados utilizado para validação final, o qual não foi implementado neste trabalho, em função da quantidade limitada dos dados que foram pré-processados.

Para a implementação da validação cruzada não foi utilizada nenhum pacote de código e/ou biblioteca do Python. A divisão do banco de dados foi realizada manualmente, sendo criados 2 dicionários da linguagem Python para cada uma das divisões do banco de dados, conforme detalhamento a seguir (os conteúdos de fala de cada uma das amostras de áudio foram detalhados na seção 3.2.):

***Split* 1:** Cada locutor foi treinado com as suas respectivas amostras de áudio: F05, F06, F07, F08, F09, F10, F11, F12, F13, F14, F15, F16, F17, F18, F19 e F20. O teste de reconhecimento foi realizado com as seguintes amostras de áudio de cada um dos locutores: F01, F02, F03 e F04.

***Split* 2:** Cada locutor foi treinado com as suas respectivas amostras de áudio: F01, F02, F03, F04, F09, F10, F11, F12, F13, F14, F15, F16, F17, F18, F19 e F20. O teste de reconhecimento foi realizado com as seguintes amostras de áudio de cada um dos locutores: F05, F06, F07 e F08.

***Split* 3:** Cada locutor foi treinado com as suas respectivas amostras de áudio: F01, F02, F03, F04, F05, F06, F07, F08, F13, F14, F15, F16, F17, F18, F19 e F20. O teste de reconhecimento foi realizado com as seguintes amostras de áudio de cada um dos locutores: F09, F10, F11 e F12.

***Split* 4:** Cada locutor foi treinado com as suas respectivas amostras de áudio: F01, F02, F03, F04, F05, F06, F07, F08, F09, F10, F11, F12, F17, F18, F19 e F20 O teste de reconhecimento foi realizado com as seguintes amostras de áudio de cada um dos locutores: F13, F14, F15 e F16.

***Split* 5:** Cada locutor foi treinado com as suas respectivas amostras de áudio: F01, F02, F03, F04, F05, F06, F07, F08, F09, F10, F11, F12, F13, F14, F15 e F16. O teste de reconhecimento foi realizado com as seguintes amostras de áudio de cada um dos locutores: F17, F18, F19 e F20.

Cabe ressaltar que no caso dos locutores impostores, uma vez que eles não possuem modelos HMM cadastrados na aplicação, estes não possuem amostras de áudio nos bancos de treinamento. Entretanto, a fim de avaliar a capacidade da aplicação em rechaçar esses impostores, nos bancos de teste, foram utilizadas as mesmas sequências de amostras de áudio utilizadas para os locutores cadastrados na aplicação.

Por fim, quando se utiliza a validação cruzada para avaliar o desempenho de uma aplicação, o valor a ser considerado paras as métricas (Acurácia, *Recall*, *Precision* e *F1-score*) devem ser as médias dos resultados obtidos em cada uma das divisões testadas (*split*) (SCIKITLEARN, 2022).

4 Resultados e Discussão

Neste capítulo são discutidos os aspectos e parâmetros técnicos que influenciam no desempenho da aplicação de reconhecimento automático de locutor e apresentados os respectivos resultados obtidos.

4.1 Aspectos e Parâmetros Técnicos que Influenciam no Desempenho da Aplicação

Conforme detalhado na seção 3.5, partindo da análise exploratória foram testados diversos valores para os parâmetros da aplicação, quais sejam: quantidade de coeficientes MFCC; quantidade de estados do HMM; quantidade de gaussianas para a modelagem da f.d.p. de emissão de símbolos associada a cada estado; e proporcionalidade entre as amostras de áudio que compõem os bancos de dados de treinamento e teste.

Cabe ressaltar que até a conclusão da etapa de implementação da Aplicação de Reconhcimento Automático de Locutor descrita na seção 3.5 ainda não havia sido utilizado nenhum modelo de teste específico e automatizado para identificar os valores ótimos para esses parâmetros. Portanto, os valores apresentados na seção 3.5 não podem ser considerados como sendo ótimos, mas sim como uma boa aproximação, em face do desempenho observado.

O mecanismo utilizado para chegar aos valores que foram adotados para cada um dos parâmetros consistiu em fixar aqueles para os quais existem referencial teórico para sustentar, como é o caso da quantidade de coeficientes MFCC, que foi fixada em 12 (do 2° ao 13°), bem como fixar valores para os parâmetros baseados em boa prática, como é o caso da proporcionalidade entre as amostras de treinamento e teste, que foi fixada em 80% e 20%, por ser uma prática comum empregada no treinamento de modelos de aprendizagem supervisionada. Entretanto, não trata-se de uma regra, haja vista que dependendo do tipo da aplicação e da abordagem de teste utilizada essa proporcionalidade pode ser alterada para uma finalidade específica.

Na sequência, após essas definições, foram testados valores para a quantidade de estados dos HMMs e da quantidade de gaussianas para modelar a f.d.p. associada a cada estado. Neste caso, foram realizados testes manuais, através dos quais observou-se que deve existir um compromisso entre essas duas quantidades, a fim de obter um bom desempenho no reconhecimento dos locutores sem tornar a aplicação lenta demais, ou seja, sem elevar demais o seu custo computacional. Na análise exploratória os modelos foram concebidos com 2 estados e 1 gaussiana. Entretanto, ao aplicar a validação cruzada o desempenho

do modelo caiu drasticamente. Sendo assim, para a realização dos testes com a versão atual da aplicação, foram adotados inicialmente 4 estados e 3 gaussianas. Na sequência foram realizados seguidos testes, onde o número de estado foi reduzido e o número de gaussianas elevado, até atingir a acurácia de 1 (cem por cento) no reconhecimento dos locutores cadastrados na aplicação, o que ocorreu para 2 estados e 6 gaussianas.

Após alcançar o valor de acurácia igual a 1 (cem por cento) considerando o cenário sem a presença de impostor, o foco do desenvolvimento voltou-se para a avaliação do desempenho da estratégia adotada para rechaçar impostores.

Neste aspecto, constatou-se que o limiar de decisão θ influencia diretamente no desempenho da aplicação, ou seja, aumentando o limiar, o sistema apresenta uma maior capacidade de rechaçar locutores impostores. Entretanto, o custo para isto é que se aumenta também o número de locutores válidos rechaçados indevidamente. Por outro lado, diminuindo o limiar de decisão a taxa de acerto dos locutores válidos atinge 1 (cem por cento). Porém, a aplicação torna-se mais permissiva a autenticar locutores impostores de maneira indevida, conforme demonstrado na avaliação do desempenho da aplicação apresentada na seção 4.2.

4.2 Avaliação do Desempenho da Aplicação

As Figuras 26, 27, 28, 29 e 30 e as Tabelas 3, 4, 5, 6 e 7, apresentam as Matrizes de Confusão e as respectivas métricas de análise de desempenho da aplicação (Acurácia, *Recall*, *Precision* e *F1-score*), considerando a utilização da técnica de validação cruzada detalhada na seção 3.6.

Dentre as métricas calculadas, a Acurácia é a que melhor representa o desempenho da aplicação, uma vez que ela abrange tanto a tarefa de reconhecimento de um locutor dentre os cadastrados quanto a tarefa de rechaçar os locutores impostores. Além disso, cabe ressaltar que, pela forma como os testes foram estruturados, para avaliar o desempenho da aplicação quanto aos locutores impostores, as métricas *Recall*, *Precision* e *F1-score* não se aplicam, uma vez que todos os impostores, quando classificados corretamente, devem ser reportados como não locutor (no caso, como 0). Sendo assim, as informações VP e FN não se aplicam a este caso, resultando em divisão por 0 para o cálculo do *Recall* e valor 0 para o *Precision*. Consequentemente, o cálculo do *F1-score* resulta em divisão por 0. Portanto, para o caso dos locutores impostores, as métricas *Recall*, *Precision* e *F1-score* não possuem significado.

Em relação aos locutores cadastrados, através dos resultados apresentados nas Tabelas 3, 4, 5, 6 e 7, podemos avaliar as métricas Acurácia, *Recall*, *Precision* e *F1-score* para cada um dos locutores de forma individual, servindo para identificar algum viés do classificador e/ou das amostras de áudio utilizadas para treinamento e teste. Neste

contexto, avaliando os resultados para os valores de limiar de decisão θ valendo -100, -25, 0, 25 e 100, é possível identificar que o locutor 7 destoa dos demais, principalmente para valores de limiar mais elevado, que é quando a aplicação está mais restritiva, elevando o seu potencial de rechaçar impostores. Nessas situações é esperado que locutores cadastrados sejam rechaçados indevidamente. Entretanto, o locutor 7 apresenta resultados abaixo dos demais de forma consistente. Como há um equilíbrio entre os demais locutores, muito provavelmente o problema está nas amostras de áudio utilizadas para teste e treinamento do HMM que representa este locutor. O valor da Acurárica deste locutor acaba por puxar para baixo a Acurácia global da aplicação.

O limiar de decisão θ influencia diretamente no desempenho da aplicação. Os resultados apresentados nas Tabelas 3, 4, 5, 6 e 7, nos mostram que, para um limiar mais alto, por exemplo $\theta = 100$, a Acurácia resultante foi 0,79 (setenta e nove por cento). Entretanto, neste caso, todos os locutores impostores foram rechaçados adequadamente, ao custo de rechaçar de maneira indevida locutores válidos. Em contrapartida, para um limiar mais baixo, por exemplo $\theta = -100$, todos os locutores cadastrados são reconhecidos adequadamente. Entretanto, neste caso, todos os cinco locutores impostores utilizados nos testes conseguiram ser validados pela aplicação indevidamente ao menos uma vez dentre os 20 testes realizados para cada um deles com amostras de áudio diferentes. Contudo, a Acurária resultante foi de 0,92 (noventa e dois por cento).

O maior valor obtido para a Acurácia foi 0,96 (noventa e seis por cento), o que ocorreu para o limiar de decisão $\theta = 0$. Neste cenário, apenas os impostores 11 e 14 conseguiram ser validados pela aplicação de forma indevida (uma única vez cada), dentre os mesmos 20 testes realizados para cada um deles com amostras de áudio diferentes. Contudo, o limiar de decisão deve ser escolhido em função das características do sistema onde a aplicação será utilizada. Em sistemas onde validar um impostor indevidamente é mais impactante do que rechaçar um locutor válido, como é o caso dos sistemas de segurança, deve-se optar por um limiar mais elevado, caso contrário um limiar mais baixo pode ser utilizado sem grandes problemas.

Com o objetivo de avaliar as quatro métricas (Acurácia, *Recall*, *Precision* e *F1-score*) de maneira a englobar tanto os locutores válidos como os locutores impostores simultaneamente, uma segunda abordagem de teste foi aplicada, onde os locutores foram avaliados em uma tabela verdade composta de apenas duas classes: locutores válidos e locutores impostores.

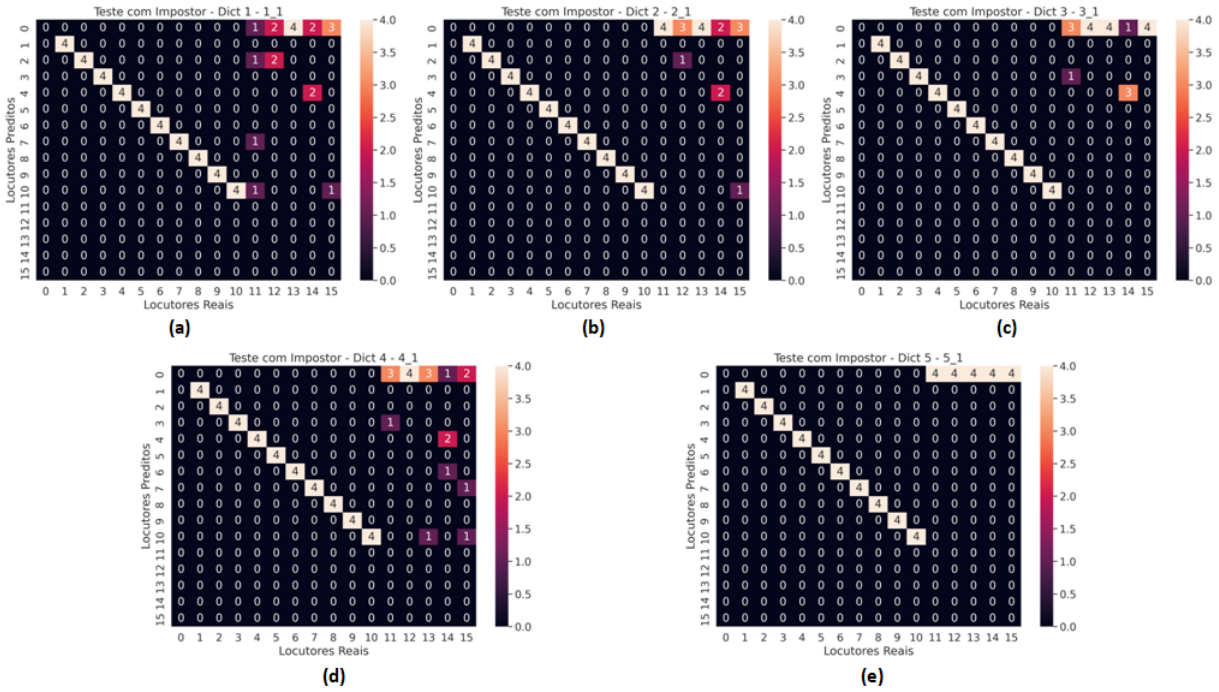
Nessa abordagem temos a representação dos resultados em uma matriz equivalente de dimensão 2x2, que nos permite avaliar, através das quatro métricas mencionadas, a capacidade da aplicação de reconhecer os locutores válidos e de rechaçar os locutores impostores de uma maneira mais ampla.

Ressalta-se que para essa abordagem também foi aplicada a validação cruzada no

mesmo formato apresentado na seção 3.6, cujos resultados são apresentados nas Figuras 31, 32, 33, 34 e 35 e nas Tabelas 8, 9, 10, 11 e 12.

Portanto, considerando os resultados obtidos para o limiar ($\theta = 0$) em que a aplicação apresenta o maior valor para a Acurácia, podemos realizar as seguintes análises:

- Acurácia igual a 0,96 (noventa e seis por cento). Este resultado é condizente com a primeira abordagem de teste e indica a média global de acertos da aplicação, englobando os locutores válidos e locutores impostores;
- *Recall* igual a 0,95 (noventa e seis por cento). Este resultado indica a proporção dos locutores classificados pela aplicação como locutores válidos em relação ao total de locutores, de fato, válidos. Em outras palavras significa dizer que dentre os 200 testes realizados com locutores válidos, 191 locutores foram reportados pela aplicação como válidos;
- *Precision* igual a 0,99 (noventa e nove por cento). Este resultado indica que a probabilidade de um locutor classificado pela aplicação como válido ser, de fato, válido é de 0,99 (noventa e nove por cento).
- *F1-score* igual a 0,97 (noventa e sete por cento). Este resultado consiste na média harmônica entre *Recall* e *Precision*, sendo mais utilizado nas análises onde se tem resultados muito discrepantes entre o *Recall* e *Precision*, o que não é o caso. Entretanto, trata-se de uma métrica de elevada importância, uma vez que possui a capacidade de equalizar o desbalanceamento entre as classes, apresentando um resultado de desempenho mais realista do que o *Recall* ou o *Precision* quando tomados individualmente. O seu uso se justifica, pois na segunda abordagem de teste, embora tenhamos explicitamente 2 classes apenas, por trás de cada uma delas temos pesos distintos, ou seja, para os locutores cadastrados temos 10 classes e são realizados 4 testes para cada uma delas, totalizando 40 testes. No caso dos locutores impostores temos 5 classes e são realizados 4 testes para cada uma delas, totalizando 20 testes. Portanto, a representatividade da classe “locutores cadastrados” é igual a 2 vezes a representatividade da classe “locutores impostores”.

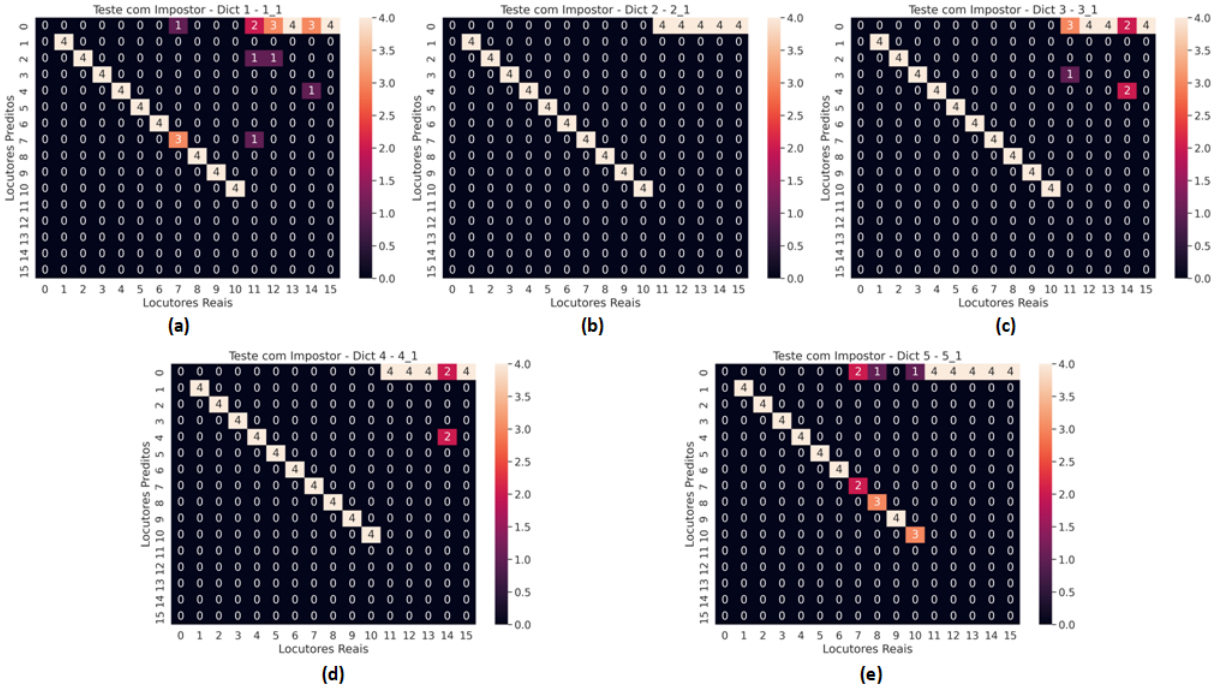


Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 26 – Matrizes de Confusão da Validação Cruzada para o Limiar $\theta = -100$.

Tabela 3 – Métricas para o limiar de $\theta = -100$.

-	Locutores Válidos										Impostores					Média
-	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	-
VP	20	20	20	20	20	20	20	20	20	20	-	-	-	-	-	-
VN	0	0	0	0	0	0	0	0	0	0	15	17	19	10	16	-
FP	0	0	0	0	0	0	0	0	0	0	5	3	1	10	4	-
FN	0	0	0	0	0	0	0	0	0	0	-	-	-	-	-	-
Acu	1	1	1	1	1	1	1	1	1	1	0,75	0,85	0,95	0,5	0,8	0,92
Rec	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	1
Pre	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	1
F1	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	1
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score																



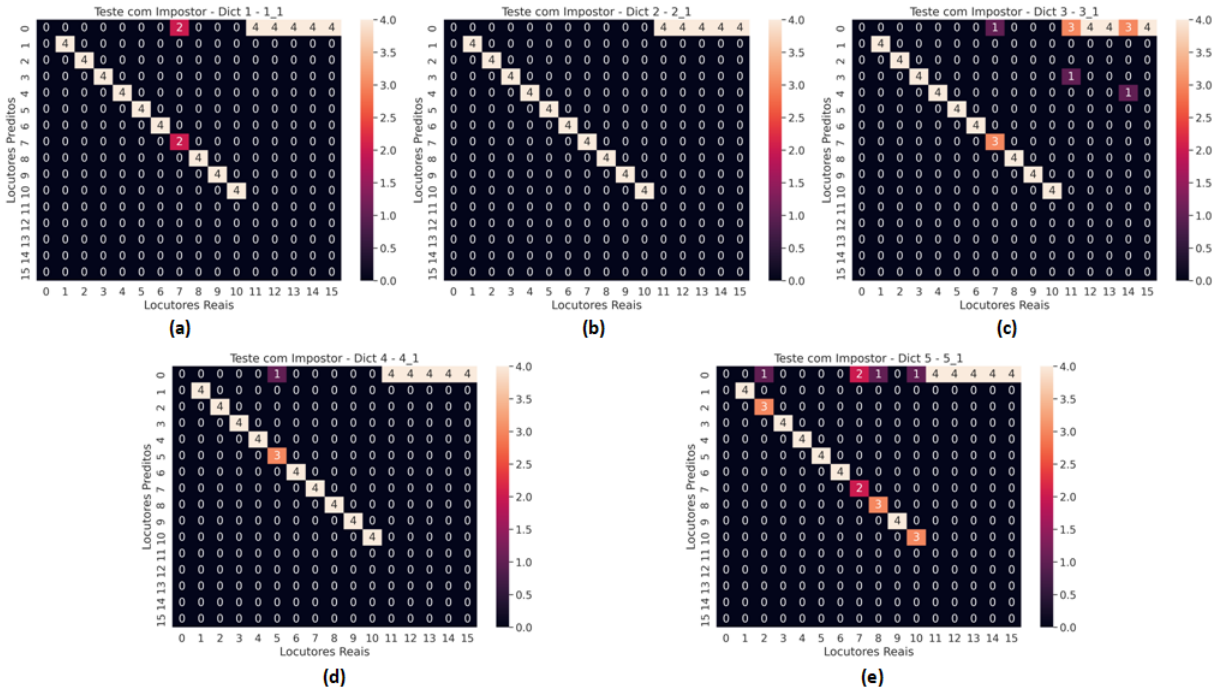
Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 27 – Matrizes de Confusão da Validação Cruzada para o Limiar $\theta = -25$.

Tabela 4 – Métricas para o limiar de $\theta = -25$.

-	Locutores Válidos										Impostores					Média
-	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	-
VP	20	20	20	20	20	20	17	19	20	19	-	-	-	-	-	-
VN	0	0	0	0	0	0	0	0	0	0	17	19	20	15	20	-
FP	0	0	0	0	0	0	0	0	0	0	3	1	0	5	0	-
FN	0	0	0	0	0	0	3	1	0	1	-	-	-	-	-	-
Acu	1	1	1	1	1	1	0,85	0,95	1	0,95	0,85	0,95	1	0,75	1	0,95
Rec	1	1	1	1	1	1	0,85	0,95	1	0,95	-	-	-	-	-	0,97
Pre	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	1
F1	1	1	1	1	1	1	0,91	0,97	1	0,97	-	-	-	-	-	0,98

Legenda:
VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo;
FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score

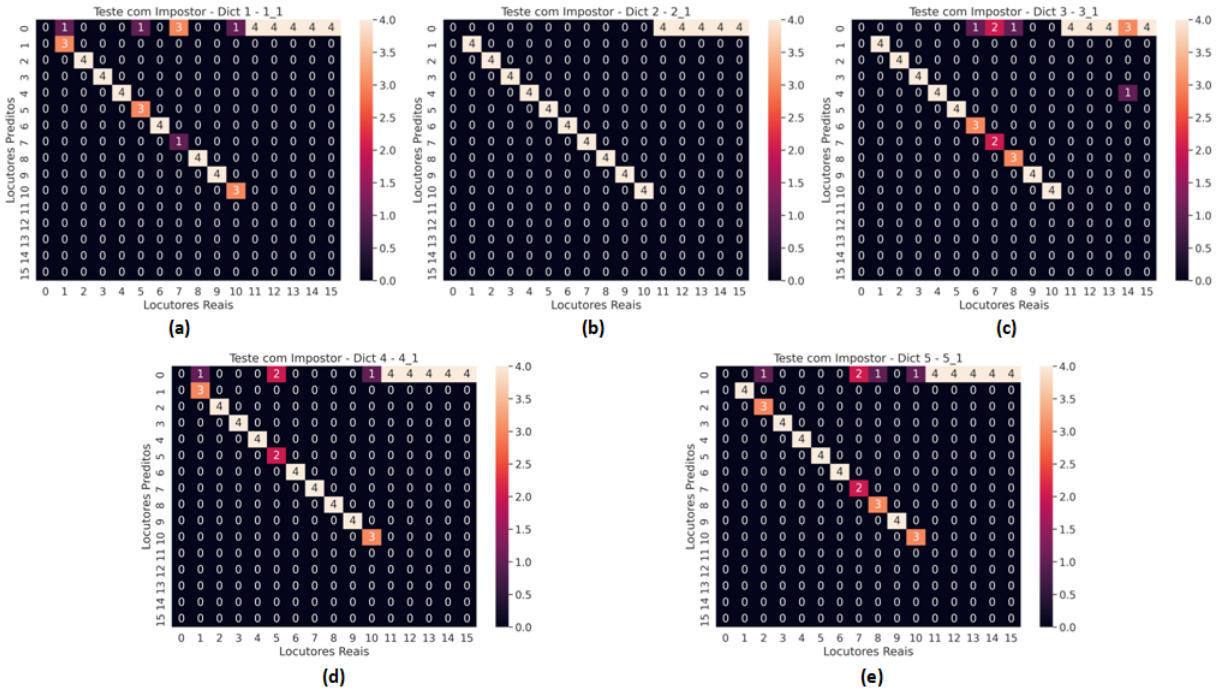


Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 28 – Matrizes de Confusão da Validação Cruzada para o Limiar $\theta = 0$.

Tabela 5 – Métricas para o limiar de $\theta = 0$.

-	Locutores Válidos										Impostores					Média
-	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	-
VP	20	19	20	20	19	20	15	19	20	19	-	-	-	-	-	-
VN	0	0	0	0	0	0	0	0	0	0	19	20	20	19	20	-
FP	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	-
FN	0	1	0	0	1	0	5	1	0	1	-	-	-	-	-	-
Acu	1	0,95	1	1	0,95	1	0,75	0,95	1	0,95	0,95	1	1	0,95	1	0,96
Rec	1	0,95	1	1	0,95	1	0,75	0,95	1	0,95	-	-	-	-	-	0,95
Pre	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	1
F1	1	0,97	1	1	0,97	1	0,85	0,97	1	0,97	-	-	-	-	-	0,97
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score																

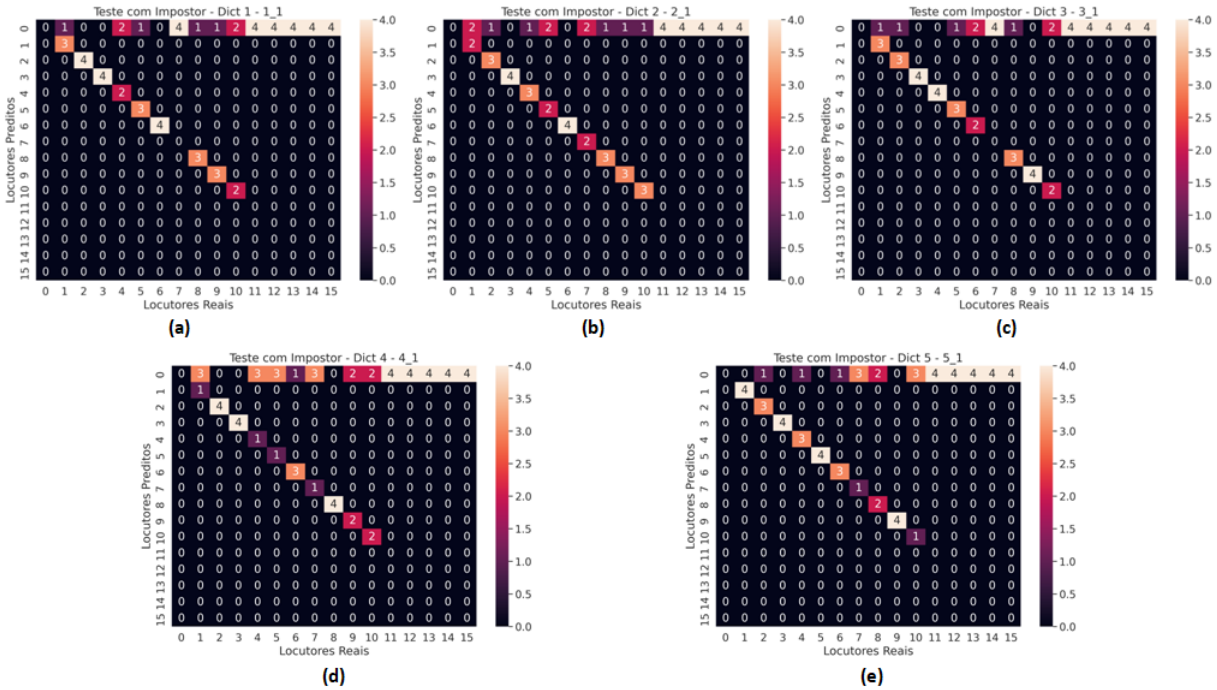


Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 29 – Matrizes de Confusão da Validação Cruzada para o Limiar $\theta = 25$.

Tabela 6 – Métricas para o limiar de $\theta = 25$.

-	Locutores Válidos										Impostores					Média
-	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	-
VP	18	19	20	20	17	19	12	18	20	17	-	-	-	-	-	-
VN	0	0	0	0	0	0	0	0	0	0	20	20	20	19	20	-
FP	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	-
FN	2	1	0	0	3	1	7	2	0	3	-	-	-	-	-	-
Acurácia:	0,9	0,95	1	1	0,85	0,95	0,63	0,9	1	0,85	1	1	1	0,95	1	0,93
Recall:	0,9	0,95	1	1	0,85	0,95	0,63	0,9	1	0,85	-	-	-	-	-	0,9
Precision:	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	1
F1-score:	0,94	0,97	1	1	0,91	0,97	0,77	0,94	1	0,91	-	-	-	-	-	0,94
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score																

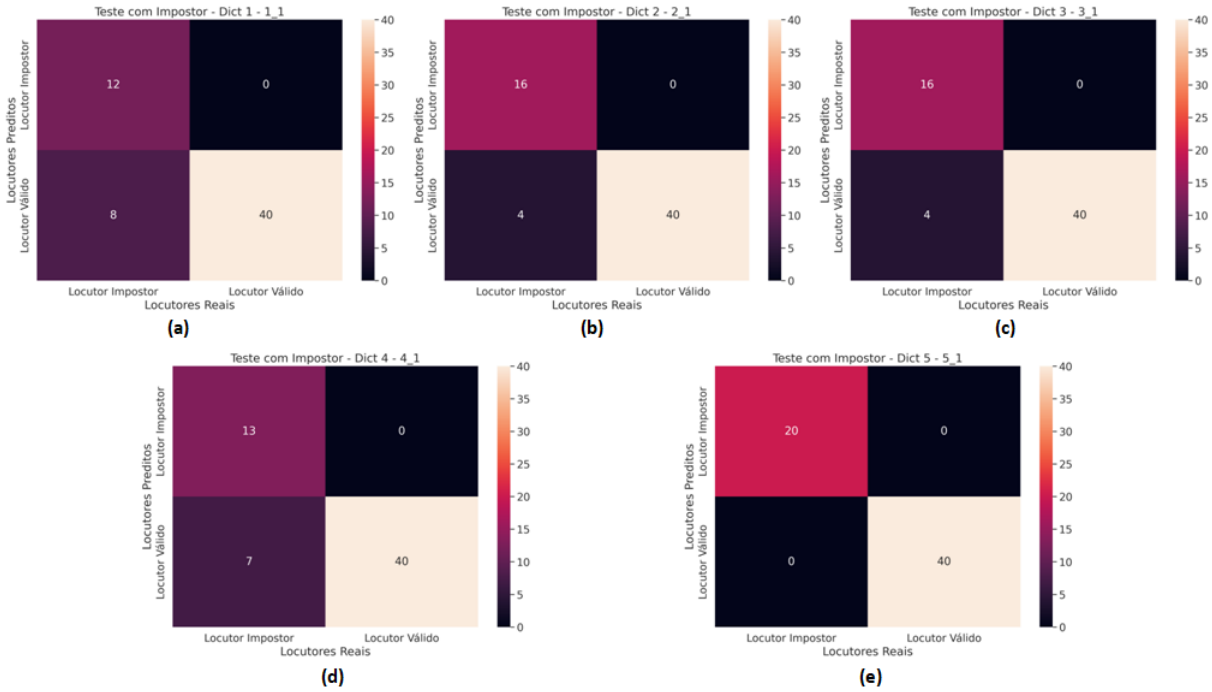


Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 30 – Matrizes de Confusão da Validação Cruzada para o Limiar $\theta = 100$.

Tabela 7 – Métricas para o limiar de $\theta = 100$.

-	Locutores Válidos										Impostores					Média
-	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	-
VP	13	17	20	13	13	16	4	15	16	10	-	-	-	-	-	-
VN	0	0	0	0	0	0	0	0	0	0	20	20	20	20	20	-
FP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-
FN	7	3	0	7	7	4	16	5	4	10	-	-	-	-	-	-
Acurácia:	0,65	0,85	1	0,65	0,65	0,8	0,2	0,75	0,8	0,5	1	1	1	1	1	0,79
Recall:	0,65	0,85	1	0,65	0,65	0,8	0,2	0,75	0,8	0,5	-	-	-	-	-	0,68
Precision:	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	1
F1-score:	0,78	0,91	1	0,78	0,78	0,88	0,33	0,85	0,88	0,66	-	-	-	-	-	0,78
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score																

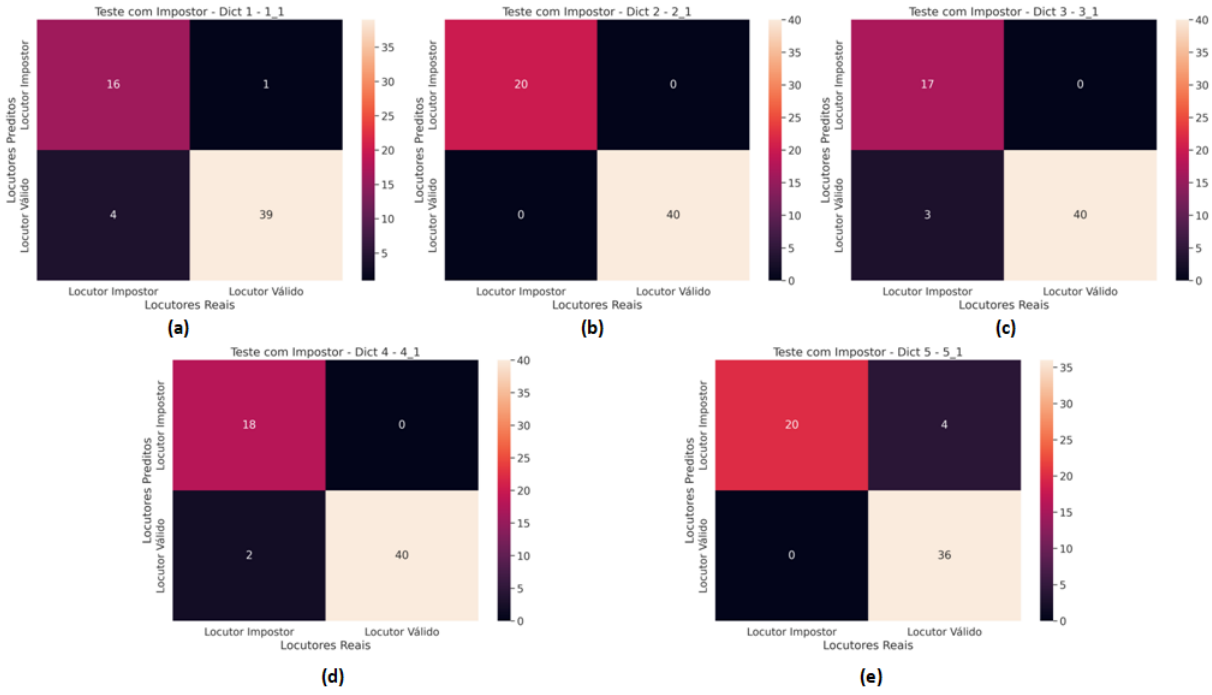


Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 31 – Matrizes de Confusão 2x2 da Validação Cruzada para o Limiar $\theta = -100$.

Tabela 8 – Métricas para o limiar de $\theta = -100$ para a Segunda Abordagem de Teste.

	Split 1	**Split 2**	**Split 3**	**Split 4**	**Split 5**	**Média**
VP	40	40	40	40	40	-
VN	12	16	16	13	20	-
FP	8	4	4	7	0	-
FN	0	0	0	0	0	-
Acurácia:	0,86	0,93	0,93	0,88	1	0,92
Recall:	1	1	1	1	1	1
Precision:	0,83	0,9	0,9	0,85	1	0,9
F1-score:	0,9	0,94	0,94	0,91	1	0,94
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score						



Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 32 – Matrizes de Confusão 2x2 da Validação Cruzada para o Limiar $\theta = -25$.

Tabela 9 – Métricas para o limiar de $\theta = -25$ para a Segunda Abordagem de Teste.

	Split 1	**Split 2**	**Split 3**	**Split 4**	**Split 5**	**Média**
VP	39	40	40	40	36	-
VN	16	20	17	18	20	-
FP	4	0	3	2	0	-
FN	1	0	0	0	4	-
Acurácia:	0,91	1	0,95	0,96	0,93	0,95
Recall:	0,97	1	1	1	0,9	0,97
Precision:	0,9	1	0,93	0,95	1	0,96
F1-score:	0,93	1	0,96	0,97	0,94	0,96
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score						

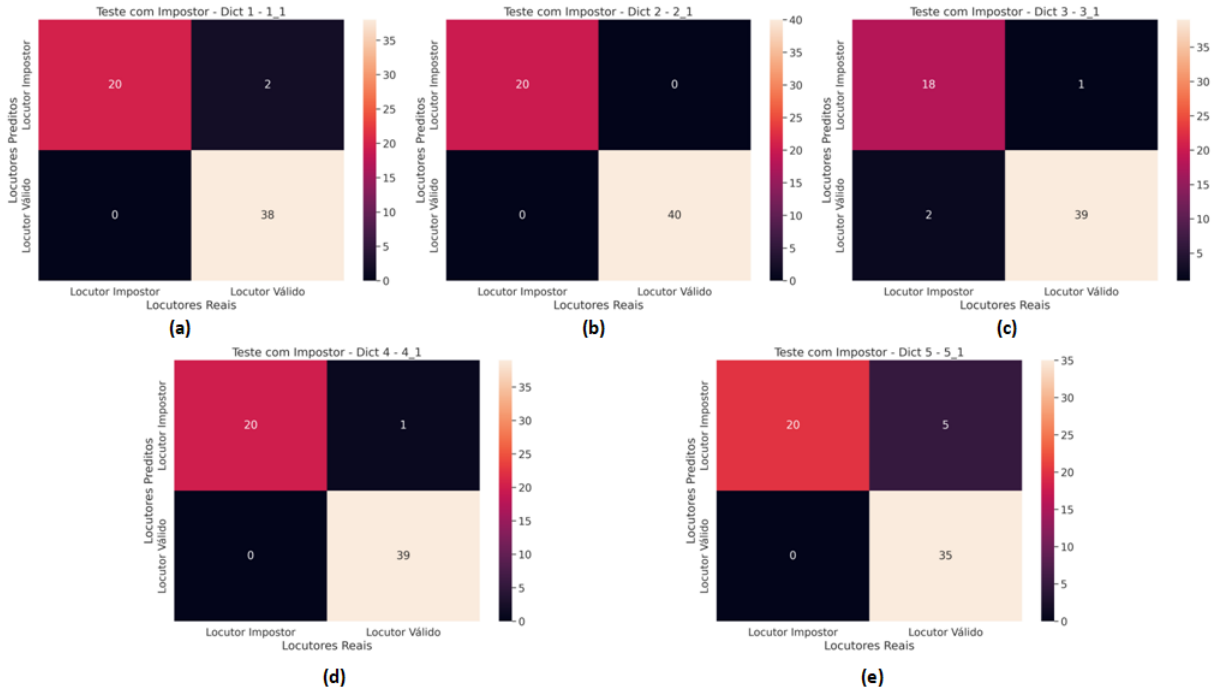
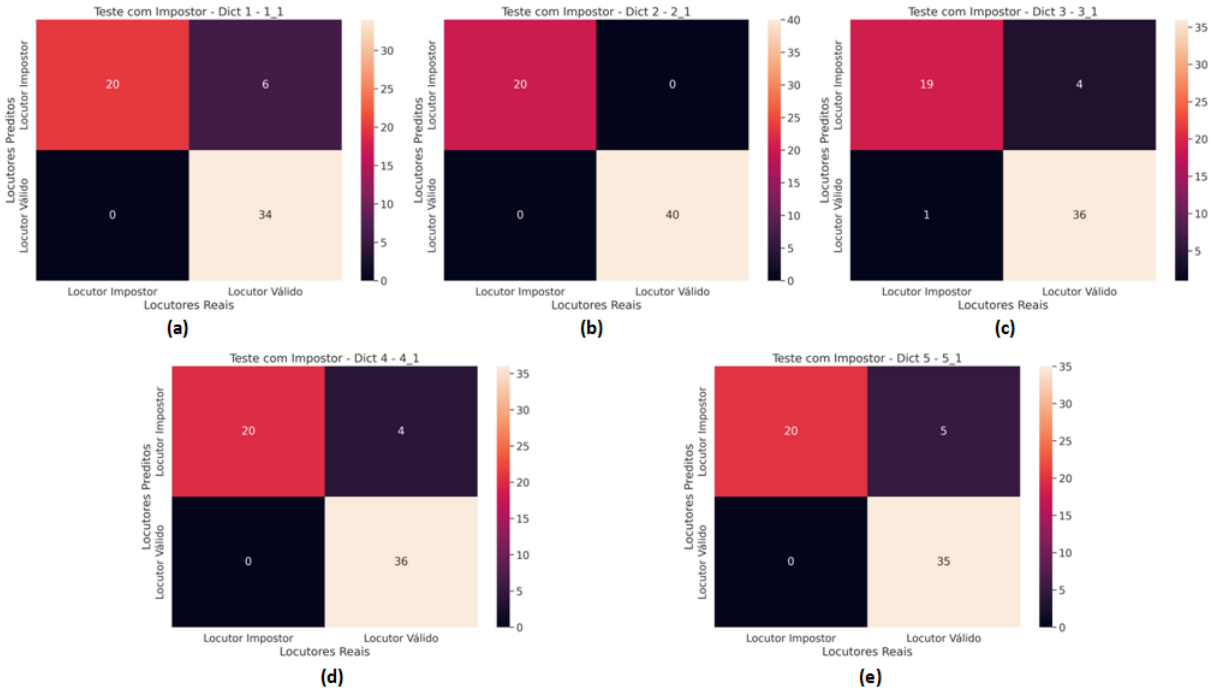


Figura 33 – Matrizes de Confusão 2x2 da Validação Cruzada para o Limiar $\theta = 0$.

Tabela 10 – Métricas para o limiar de $\theta = 0$ para a Segunda Abordagem de Teste.

	Split 1	**Split 2**	**Split 3**	**Split 4**	**Split 5**	**Média**
VP	38	40	39	39	35	-
VN	20	20	18	20	20	-
FP	0	0	2	0	0	-
FN	2	0	1	1	5	-
Acurácia:	0,96	1	0,95	0,98	0,91	0,96
Recall:	0,95	1	0,97	0,97	0,87	0,95
Precision:	1	1	0,95	1	1	0,99
F1-score:	0,97	1	0,95	0,98	0,93	0,97
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score						

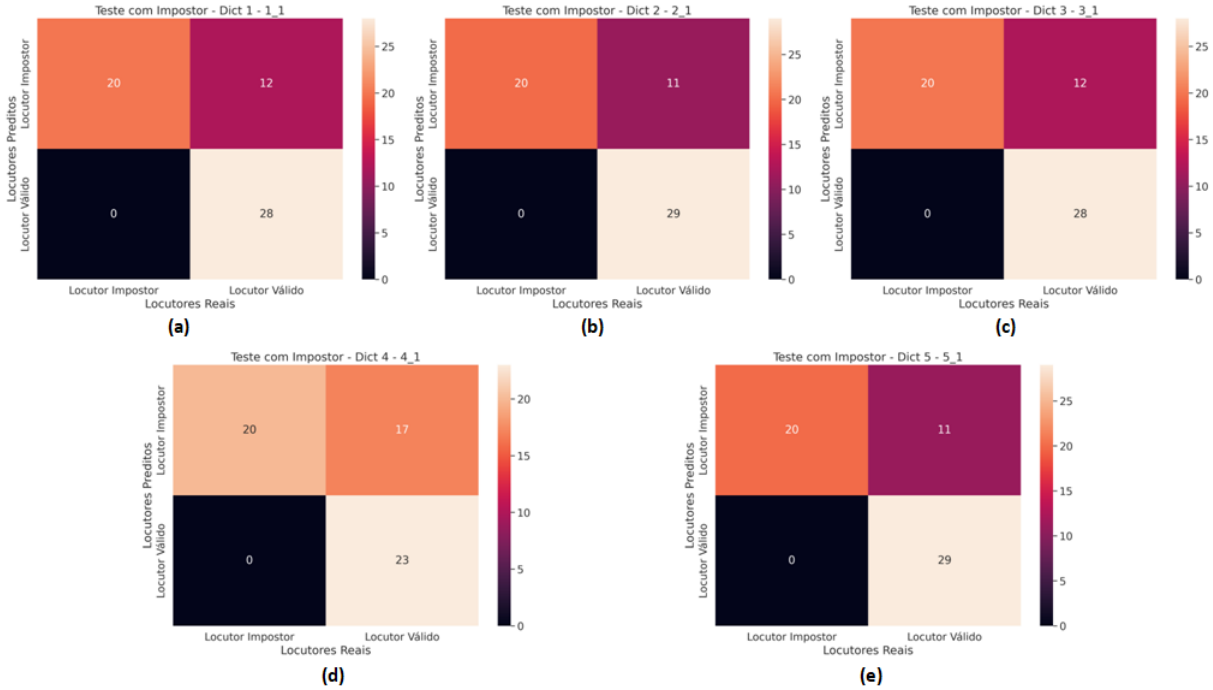


Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 34 – Matrizes de Confusão 2x2 da Validação Cruzada para o Limiar $\theta = 25$.

Tabela 11 – Métricas para o limiar de $\theta = 25$ para a Segunda Abordagem de Teste.

	Split 1	**Split 2**	**Split 3**	**Split 4**	**Split 5**	**Média**
VP	34	40	36	36	35	-
VN	20	20	19	20	20	-
FP	0	0	1	0	0	-
FN	6	0	4	4	5	-
Acurácia:	0,9	1	0,91	0,93	0,91	0,93
Recall:	0,85	1	0,9	0,9	0,87	0,9
Precision:	1	1	0,97	1	1	0,99
F1-score:	0,91	1	0,93	0,94	0,93	0,94
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score						



Legenda: (a) – Split 1; **(b)** – Split 2; **(c)** – Split 3; **(d)** – Split 4; **(e)** – Split 5.

Figura 35 – Matrizes de Confusão 2x2 da Validação Cruzada para o Limiar $\theta = 100$.

Tabela 12 – Métricas para o limiar de $\theta = 100$ para a Segunda Abordagem de Teste.

	Split 1	**Split 2**	**Split 3**	**Split 4**	**Split 5**	**Média**
VP	28	29	28	23	29	-
VN	20	20	20	20	20	-
FP	0	0	0	0	0	-
FN	12	11	12	17	11	-
Acurácia:	0,8	0,81	0,8	0,71	0,81	0,79
Recall:	0,7	0,72	0,7	0,57	0,72	0,68
Precision:	1	1	1	1	1	1
F1-score:	0,82	0,83	0,82	0,72	0,83	0,8
Legenda: VP - Verdadeiro Positivo; VN - Verdadeiro Negativo; FP - Falso Positivo; FN - Falso Negativo; Acu - Acurácia; Rec - Recall; Pre - Precision; e F1 - F1-score						

4.3 Sensibilidade da Aplicação de Reconhecimento Automático de Locutor aos Parâmetros do HMM

Na seção 2.5 foi introduzido que a complexidade computacional de um HMM, que utiliza algoritmos *Forward & Backward* para os problemas de treinamento e avaliação, é da ordem de N^2M operações, onde N é o número de estados ocultos e M é o número de observações. Ainda na referida seção foi introduzido que o número de estados de um HMM está muito mais relacionado com a complexidade computacional do modelo do que com a qualidade dos resultados apresentados.

A fim de confirmar a previsão teórica quanto à complexidade computacional e o desempenho dos modelos HMMs empregados na Aplicação de Reconhecimento Automático de Locutor foram realizados testes automatizados para mensurar o tempo necessário para o treinamento do modelo de fundo (UBM), o tempo médio necessário para o treinamento dos modelos individuais que representam cada um dos locutores cadastrados na aplicação (GMM), o tempo médio necessário para o reconhecimento de um locutor a partir de um vetor de observações M, e a Acurácia da Aplicação de Reconhecimento Automático de Locutor, em função do número de estados N e do número de gaussianas utilizado para modelar a f.d.p. de emissão de símbolos associada a cada estado.

A Figura 36 apresenta o resultado do teste realizado para mensurar o tempo de treinamento do modelo de fundo (UBM) em função do número de estados N e do número de gaussianas utilizadas para modelar a f.d.p. de emissão de símbolos associada a cada estado.

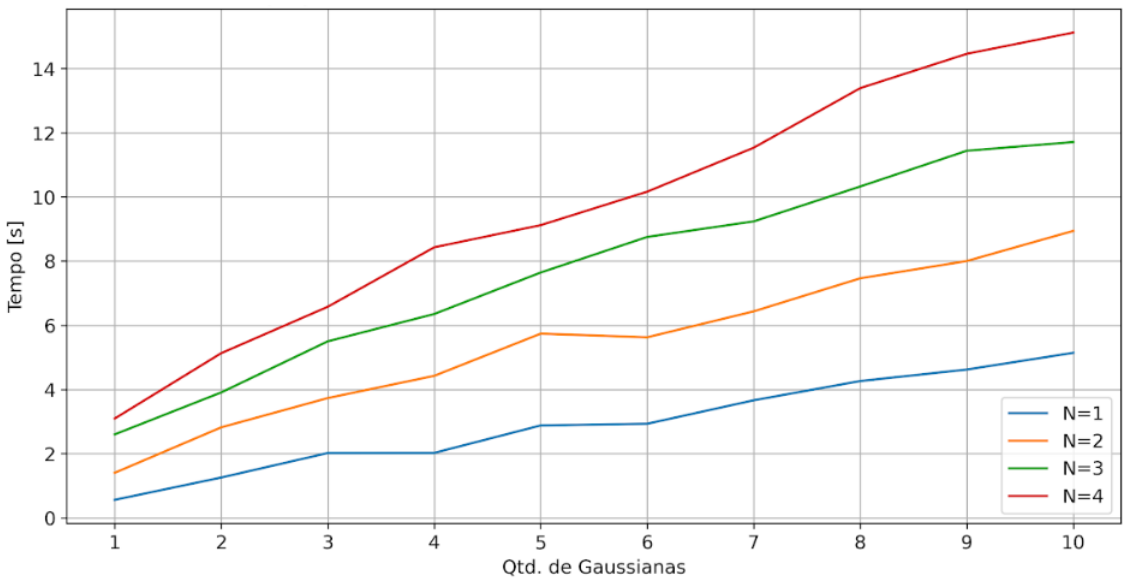


Figura 36 – Tempo de Treinamento do UBM vs Número de Estados vs Número de Gaussianas.

A Figura 37 apresenta o resultado do teste realizado para mensurar o tempo

médio de treinamento dos modelos individuais que representam cada um dos locutores cadastrados na aplicação (GMM) em função do número de estados N e do número de gaussianas utilizadas para modelar a f.d.p. de emissão de símbolos associada a cada estado.

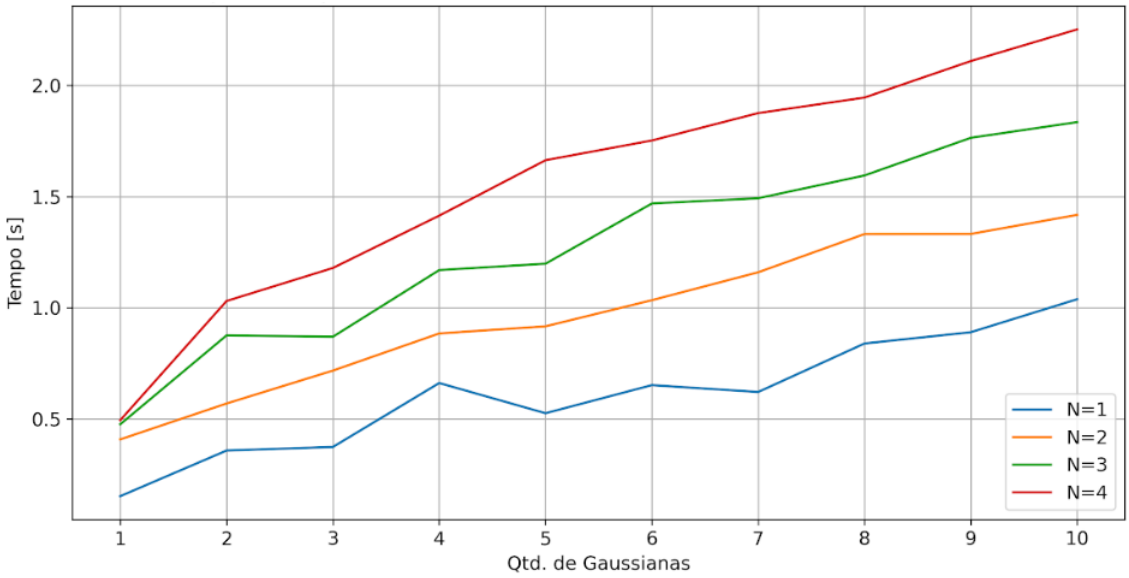


Figura 37 – Tempo Médio de Treinamento de um GMM vs Número de Estados vs Número de Gaussianas.

Avaliando os resultados apresentados nas Figuras 36 e 37 constata-se que, de fato, a complexidade computacional dos modelos aumenta tanto em função do número de estados N quanto em função do número de gaussianas utilizadas para modelar as f.d.p. de emissão de símbolos associada a cada estado. Cabe ressaltar que neste teste a complexidade computacional está sendo inferida através do tempo necessário para concluir o treinamento do modelo, uma vez que o tempo é diretamente proporcional ao número de operações necessárias para o modelo convergir.

A Figura 38 apresenta o resultado do teste realizado para mensurar o tempo médio de reconhecimento de um locutor pela Aplicação de Reconhecimento Automático de Locutor em função do número de estados N e do número de gaussianas utilizadas para modelar a f.d.p. de emissão de símbolos associada a cada estado.

Analogamente ao que ocorre na etapa de treinamento, o tempo médio de reconhecimento também nos mostra que a complexidade computacional dos modelos aumenta tanto em função do número de estados N quanto em função do número de gaussianas utilizadas para modelar as f.d.p. de emissão de símbolos associada a cada estado. Neste caso em especial, para $N = 4$ o tempo de reconhecimento acaba degenerando quando o número de gaussianas é maior do que 9. Outro aspecto importante que pode ser observado é que o tempo de reconhecimento é extremamente pequeno, evidenciando uma das principais características de desempenho dos modelos de reconhecimento de padrões probabilísticos.

Por fim, a Figura 39 apresenta o resultado do teste realizado para avaliar o

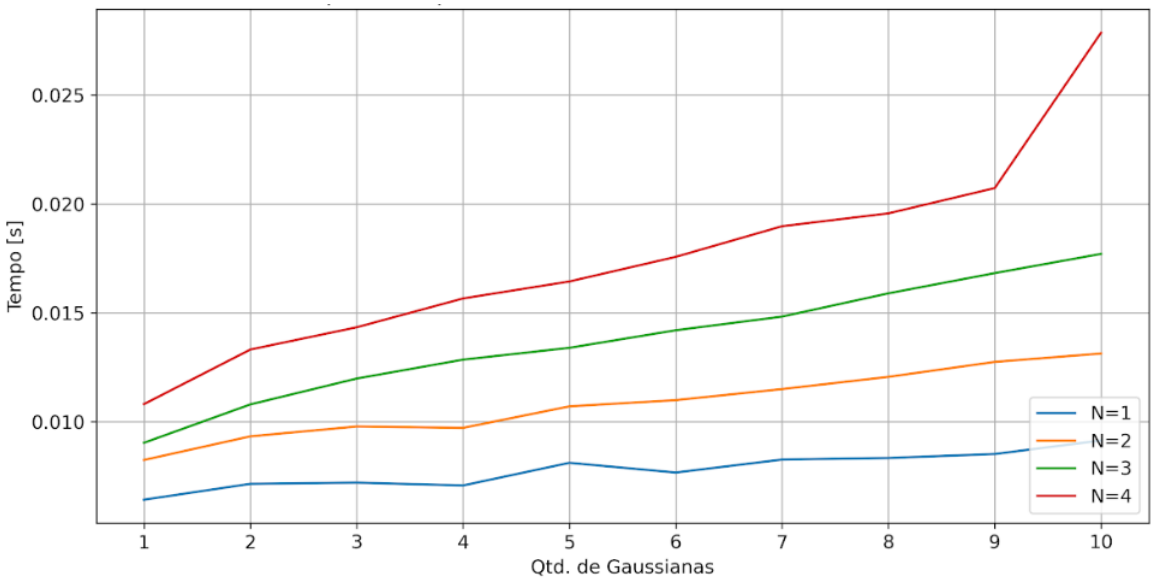


Figura 38 – Tempo Medio para Reconhecimento de um Locutor vs Número de Estados vs Número de Gaussianas.

desempenho da Aplicação de Reconhecimento Automático de Locutor em função do número de estados N e do número de gaussianas utilizadas para modelar a f.d.p. de emissão de símbolos associada a cada estado.

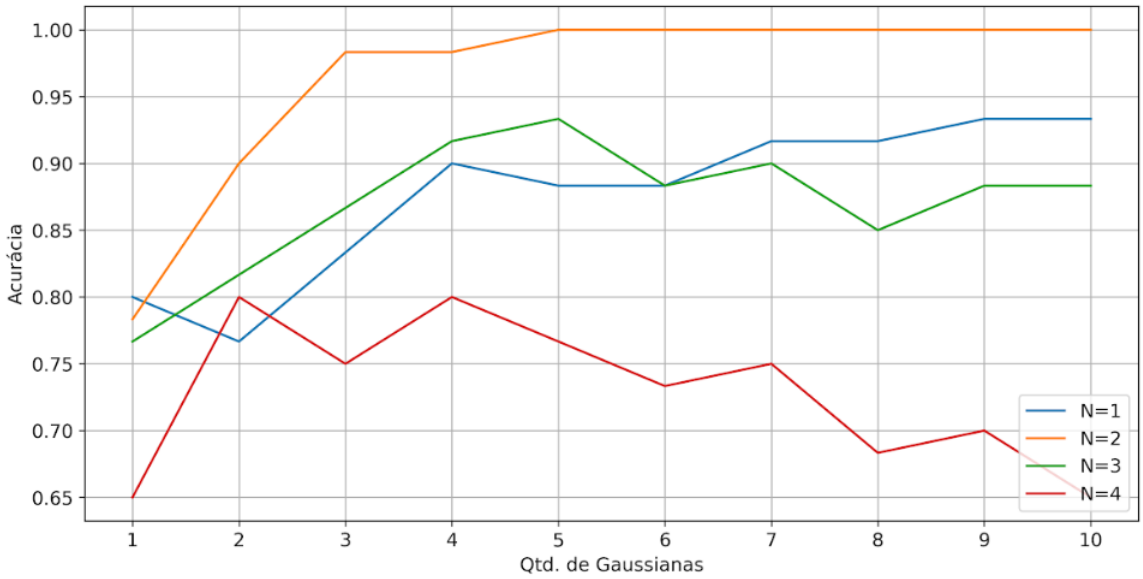


Figura 39 – Acurácia vs Número de Estados vs Número de Gaussianas.

Os resultados apresentados na Figura 39 nos mostram que, de maneira geral, o desempenho da aplicação (no caso, avaliado em função da Acurácia) não é melhorado com o aumento indiscriminado do número de estados N e do número de gaussianas utilizadas para modelar as f.d.p. de emissão de símbolos associada a cada estado. Neste teste, fica evidente que o melhor resultado, do ponto de vista da Acurácia, foi obtido para $N = 2$ e número de gaussianas maior do que 5, mantendo-se estável até o limite de valores testado.

Para as demais combinações de números de estados e número de gaussianas, a Acurácia resultante oscila bastante, ficando sempre abaixo do resultado obtido para $N = 2$.

Contudo, os resultados obtidos nos testes confirmam a previsão teórica quanto à complexidade computacional e ao desempenho da Aplicação de Reconhecimento Automático de Locutor em função do número de estados N e do número de observações M, bem como confirmam que a escolha de $N = 2$ e número de gaussianas igual a 6 utilizados no desenvolvimento apresentado na seção 3.5, de fato, foram corretos e apresentam a melhor relação custo-benefício para o desempenho do modelo em relação à sua complexidade computacional. Cabe ressaltar que o objetivo dos testes foi simplesmente avaliar o comportamento dos modelos HMM e, consequentemente, da Aplicação de Reconhecimento Automático de Locutor, em função dos seus parâmetros configuráveis N e o número de gaussianas para modelar as f.d.p., considerando uma quantidade fixa de observações. Portanto, foi escolhida e fixada a divisão do banco de dados entre amostras de treinamento e testes que resultou no melhor desempenho da aplicação (i.e., maior valor de Acurácia) dentre as divisões utilizadas na validação cruzada, que no caso o *Split* 2. Também é importante ressaltar que o limiar de decisão utilizado nos teste foi $\theta = 0$.

5 Conclusões

A solução obtida a partir da metodologia idealizada para o desenvolvimento da Aplicação de Reconhecimento Automático de Locutor se mostrou efetiva e condizente com os objetivos traçados, bem como alinhada com o previsto no referencial teórico utilizado para embasar a realização deste trabalho, ou seja, uma abordagem clássica de aprendizado de máquina supervisionado e de baixo processamento, utilizando, para a extração de características do sinal de voz, a técnica MFCC e, para a modelagem do classificador, a técnica que emprega HMM, com f.d.p. de emissão de símbolos modelada por misturas de gaussianas (HMM-GMM).

Em relação ao desempenho da aplicação, os resultados obtidos são considerados satisfatórios. Evidente que a quantidade de dados utilizada para a realização dos testes foi limitada e que para uma validação desses resultados de maneira mais precisa seria necessário testar a aplicação sobre um banco de dados maior. Entretanto, a utilização da validação cruzada para a realização dos testes indica que os resultados são consistentes e que a solução desenvolvida é de elevado potencial de utilização prática.

O fato de a abordagem utilizada para o desenvolvimento da Aplicação de Reconhecimento Automático de Locutor ser de baixo processamento, associado aos excelentes resultados obtidos, nos permite vislumbrar variadas situações práticas de utilização da aplicação, tais como: em instalações de baixa segurança, desbloqueios de tela de celular, tecnologias assistivas e plataformas embarcadas voltadas para veículos automotores, onde, além de controlar o acesso ao veículo (abertura das portas), seria possível reconhecer o motorista através da sua voz, desde que previamente cadastrado, e entregar essa informação à central de controle do veículo responsável pela sua partida ou customização de acessórios, como posição dos espelhos retrovisores, altura do assento, modo de direção, dentre outras funcionalidades.

Além das implementações práticas para a Aplicação de Reconhecimento Automático de Locutor em plataformas móveis ou embarcadas apontadas como possíveis desdobramentos futuros deste trabalho, outras perspectivas consistem no aperfeiçoamento da aplicação, a fim de obter a máxima capacidade de restringir a autenticação de locutores impostores e diminuir a taxa de rejeição de locutores válidos. Para isso poderiam ser propostas análises acerca das etapas que foram abstraídas neste trabalho, ou seja, a filtragem e detecção de início e fim dos áudios em uma etapa anterior à etapa de extração dos coeficientes MFCC. Outras possibilidades seriam: i) avaliar novos valores no processo de janelamento do sinal na etapa de extração dos coeficientes MFCC; ii) limitar o tamanho dos áudios utilizados a um mesmo valor para todos os locutores; ou iii) avaliar a utilização dos parâmetros Delta

e Delta-Delta, que consistem na primeira derivada e na segunda derivada dos coeficientes MFCC, respectivamente.

Portanto, em face da metodologia apresentada, dos resultados obtidos e das perspectivas de trabalhos futuros apontadas como possíveis desdobramentos deste trabalho de graduação, o autor espera humildemente ter contribuído para a comunidade acadêmica e amantes de tecnologia que tenham interesse pelo tema de biometria por voz e que possam utilizar este trabalho como ponto de partida para trabalhos futuros, em especial aqueles voltados para implementações práticas.

Referências

ALVEZ, G. d. L. A.; CALTABIANO, C. C.; BOLZAN, M. J. A. Análise de padrões de voz através da transformada em ondeletas. *XIII Encontro Latino Americano de Iniciação Científica e IX Encontro Latino Americano de Pós-Graduação – Universidade do Vale do Paraíba*, 2009. Citado na página 1.

ANDRADE, M. A. R. d. Fundamentos de modelos de markov escondidos (hmm). *Revista Militar de Ciência e Tecnologia*, Vol. XVII, 2000. Citado 2 vezes nas páginas 11 e 12.

AWARE. *Autenticação de voz*. 2021. Disponível em: <https://www.aware.com/pt/autenticacao-de-voz/>. Acesso em: 21 jul. 2021. Citado na página 3.

BIOMETRIA. *In: DICIO, Dicionário Online de Português. Porto: 7Graus,*. 2020. Disponível em: <https://www.dicio.com.br/biometria/>. Acesso em: 19 jul. 2021. Citado na página 1.

CRIPTOGRAPHY, P. *Mel Frequency Cepstral Coefficient (MFCC) tutorial*. 2012. Practical Criptography. Disponível em: <http://www.practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/>. Acesso em: 19 jul. 2021. Citado 2 vezes nas páginas 8 e 10.

DAVIS, S.; MERMELSTEIN, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 28, n. 4, p. 357–366, 1980. Citado na página 7.

FACHINI, A. R.; HEINEN, M. R. Aplicação de mfcc para modelar sons de instrumentos musicais. 2016. Disponível em: <http://abricom.org.br/wp-content/uploads/2016/03/bricsccicbic2013_submission_55.pdf>. Acesso em: 29 jul. 2021. Citado 3 vezes nas páginas 6, 8 e 9.

FECHINE, J. M. *Verificação de Locutor Utilizando Modelos de Markov Escondidos (HMMs) de Densidades Discretas.* Dissertação (Mestrado) — Universidade Federal da Paraíba, 1994. Citado 7 vezes nas páginas 6, 1, 4, 5, 10, 12 e 32.

GUANGA, A. *Understand Classification Performance Metrics. Becoming Human: Artificial Intelligence Magazine*. 2018. Disponível em: <https://becominghuman.ai/understand-classification-performance-metrics-cad56f2da3aa>. Acesso em: 19 oct. 2018. Citado 3 vezes nas páginas 6, 16 e 17.

HMMLEARN. *hmmlearn 0.2.6*. 2021. Disponível em: <https://pypi.org/project/hmmlearn/>. Acesso em: 21 nov. 2021. Citado na página 22.

JURAFSKY, D.; MARTIN, J. H. *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. [S.l.]: Pearson Education, 2009. Citado 2 vezes nas páginas 6 e 9.

KUINCHTNER, D. Predição do mercado de ações usando hidden markov model. 2018. Artigo de conclusão de curso (Bacharel em Ciência da Computação). Curso de Ciência da Computação. Citado 4 vezes nas páginas 6, 10, 11 e 12.

LYONS, J. *'python_speech_features' Official Documentation*. 2013. Python_speech_features. Disponível em: <https://python-speech-features.readthedocs.io/en/latest/>. Acesso em: 19 jul. 2021. Citado na página 22.

NETO, A. F. *Modelo de Autenticação Aplicado a Sistemas de Verificação de Locutor*. Tese (Doutorado) — Universidade Federal de Minas Gerais, 2018. Citado 5 vezes nas páginas 6, 27, 32, 33 e 35.

NETO, A. F.; SILVA, A.; YEHIA, H. Corpus cefala-1: Base de dados audiovisual de locutores para estudos de biometria, fonética e fonologia. *Revista de Estudos da Linguagem, v. 27, n. 1*, 2019. Citado 3 vezes nas páginas 3, 7 e 21.

PINHEIRO, J. M. *Biometria nas Redes dos ISP's: Um passo à frente em segurança – Revista ISPMAIS*. 2019. Disponível em: <https://www.ispblog.com.br/2019/08/23/biometria-nas-redes-dos-isps-um-passo-a-frente-em-seguranca/>. Acesso em: 09 abr. 2022. Citado 2 vezes nas páginas 8 e 2.

REYNOLDS, D.; QUATIERI, T.; DUNN, R. B. Speaker verification using adapted gaussian mixture models. digital signal processing, 10(1):19–41. 2000. Citado 3 vezes nas páginas 6, 33 e 35.

SCIKITLEARN. *scikit-learn 1.0.2*. 2022. Disponível em: <https://scikit-learn.org/stable/modules/cross_validation>. Acesso em: 29 mar. 2022. Citado 4 vezes nas páginas 6, 36, 37 e 38.

SEDRA, A. S.; SMITH KENNETH, C. *Microeletrônica*. [S.l.]: Pearson Education do Brasil, 2000. Citado na página 6.

SILVA, W.; GOMES, F. Reconhecimento de voz para autenticacÇÃo biomÉtrica utilizando mÁquinas de vetores de suporte e os coeficientes mel-cepstrais. 2015. Citado na página 2.

SILVA, W.; SERRA, G. *Inteligência Computacional Aplicada ao Reconhecimento de Voz*. 15, 38 p. Tese (Doutorado) — Universidade Federal do Maranhão, 2019. Citado na página 6.

TODOR, G.; NIKOS, F.; GEORGE, K. Comparative evaluation of various mfcc implementations on the speaker verification task. *in 10th International Conference on Speech and Computer (SPECOM 2005)*, v. 1, p. 191–194, 2005. Citado na página 8.