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

# Extração e análise de características de áudios para reconhecimento de locutores

**Thiago Henrique Gomes Panini**

**Santo André**

**Julho de 2021**

Thiago Henrique Gomes Panini

# Extração e análise de características de áudios para reconhecimento de locutores

**Trabalho de Graduação** apresentado para conclusão da Graduação em Engenharia de Informação, como parte dos requisitos necessários para a obtenção do Título Bacharel em Engenharia de Informação.

Universidade Federal do ABC

Orientador: Murilo Bellezoni

Santo André

Julho de 2021

Thiago Henrique Gomes Panini

# Extração e análise de características de áudios para reconhecimento de locutores

**Trabalho de Graduação** apresentado para conclusão da Graduação em Engenharia de Informação, como parte dos requisitos necessários para a obtenção do Título Bacharel em Engenharia de Informação.

Santo André, 21 de Julho de 2021:

---

**Murilo Bellezoni**
Orientador

---

**Kenji Nose Filho**
Convidado 1

---

**Tito Caco Curimbaba Spadini**
Convidado 2

Santo André
Julho de 2021

*Em primeiro lugar a Deus pela graça dada e a força para superar obstáculos em tempos difíceis. Ao meus pais, pois são a base de tudo que sou hoje e, por isso, possuem minha eterna gratidão em qualquer situação. Aos meus professores pelo dom do ensinamento e direcionamento acadêmico crucial dentro de minha formação técnica e pessoal. Aos meus amigos por toda construção sociológica em torno do meu ser. Sem eles, nada seria possível.*

# Agradecimentos

Gostaria de agradecer à Universidade Federal do ABC por toda a estrutura fornecida e pela imensa contribuição em minha formação acadêmica. A árdua jornada é especialmente recompensada por blocos fundamentais intrínsecos e consolidados em meu ser, tornando-me uma pessoa extremamente melhor dentro das diretrizes técnicas, sociais e humanas. A abrangência de citações neste parágrafo envolve todos os profissionais desta instituição que, de certa forma, provam diariamente que o vasto mundo acadêmico é constituído por uma riqueza de essências distintas que, unidas por um propósito nobre, permitem a melhor moldagem possível de profissionais e seres humanos capazes de alcançar os mais grandiosos obtivos.

Ainda neste universo, deixo um agradecimento especial ao Professor Orientador Murilo Loiola que, muito além da excelência no direcionamento deste trabalho de graduação, tornou-se um profissional exemplo já em aulas dadas para matérias do curso de Engenharia de Informação. Sua extrema dedicação na função de professor abriu portas para que eu pudesse absorver e ter como prática o dom de ensinar também àqueles que solicitam meu auxílio.

Por fim, agradeço aos meus pais que, mesmo nos tempos mais difíceis, sempre fizeram de tudo para que eu alcançasse meus objetivos acadêmicos e profissionais. Saibam que seu apoio foi e continua sendo imensurável dentro de tudo que me faz, diariamente, alguém melhor e capaz de retribuir o amor e a força que me são dadas. Que um dia eu consiga ser o exemplo que vocês se tornaram.

# Resumo

A voz humana é um dos elementos sonoros de maior importância em um cenário cotidiano de conversação. O reconhecimento de locutores a partir da fala é uma das formas mais ricas de utilizar estes sinais biológicos para propósitos tecnológicos relacionados à biometria por voz, ao acionamento de dispositivos ou mesmo a aplicações forenses. Entretanto, os fatores que permitem a construção de dada inteligência giram em torno da extração de características fundamentais de sinais falados dentro de tratamentos específicos voltados à modelagem de dados. Dito isso, este trabalho considerou, como ponto de partida, a construção de uma base de dados a partir da gravação de sinais de áudio de locutores definidos previamente. A partir deste ponto, foram utilizados conceitos técnicos e implementações já existentes para propor um fluxo automático de extração e análise de características de sinais de áudio essenciais dentro do contexto de reconhecimento de locutores. O resultado obtido pelo *pipeline* de preparação dos sinais de áudio foi utilizado como entrada para o treinamento de modelos de aprendizado de máquina, tais como Árvores de Decisão, Florestas Aleatórias e LightGBM em uma condição de aprendizado supervisionado, permitindo assim a obtenção de um modelo de classificação eficaz capaz de realizar a identificação de locutores a partir novos sinais de áudio. Ao final do processo, foi proposta uma análise específica nas principais características de sinais de áudio com maior relevância para a identificação de locutores e, entre elas, é possível citar a grande representatividade dos coeficientes MFCCs, espectrograma, centroide espectral e envelope de amplitude. Como motor principal de aplicação, a linguagem Python foi utilizada em conjunto com a biblioteca LibROSA.

**Palavra Chave**: Voz, aprendizado de máquina, reconhecimento de locutores, classificação multiclasse, características de áudio.

# Abstract

The human voice is one of the most important element in a daily conversation scenario. The speaker recognition task is one of the ways to use these biological signals that is extremely easy to generate for technological purposes related to voice biometrics, device activation or even for forensic applications. However, the factors that allow the construction of a given intelligence revolve around the extraction of fundamental characteristics of spoken signals within specific treatments aimed at data modeling. That said, this work considered, as a starting point, the creation of a database through audio recording from previously defined speakers. From this point onwards, technical concepts and existing implementations were used to propose an automatic flow of extraction and analysis of essential audio signal characteristics within the context of speaker recognition. The result obtained by the audio feature extraction pipeline was used as input for machine learning models like Decision Trees, Random Forest and LightGBM in a supervised learning strategy, thus allowing to obtain an efficient classification model capable of carrying out the identification of speakers from new audio signals. At the end of the process, a specific analysis was proposed on the main characteristics of audio signals with greater relevance for the identification of speakers and, among them, it is possible to cite the great representation of the MFCC coefficients, spectrogram, spectral centroid and amplitude envelope. As the main application engine, the Python language was used in conjunction with the LibROSA library.

**Keywords**: Voice, machine learning, speaker recognition, multiclass classification, audio features.

# Lista de abreviaturas e siglas

| | |
|---|---|
| BER | Band Energy Ratio |
| DCF | Detection Cost Function |
| DCT | Discrete Cosine Transform |
| DFT | Discrete Fourier Transform |
| EER | Equal Error Rate |
| FAR | False Acceptance Ratio |
| FRR | False Rejection Error |
| FTT | Fast Fourier Transform |
| GMM | Gaussian Mixture Models |
| MFCC | Mel-Frequency Cepstral Coefficients |
| NIST | National Institute of Standards and Techniques |
| STFT | Short-Time Fourier Transform |
| UBM | Universal Background Model |
| ZCR | Zero-Crossing Rate |

# Lista de ilustrações

# Lista de tabelas

# Sumário

# 1 Introdução

A fala é uma das mais ricas ferramentas de comunicação do ser humano. Historicamente, a partir dela, diversas outras relações puderam ser desenvolvidas em uma grande aceleração do processo evolutivo, sendo este notado em áreas comerciais, pessoais, econômicas, artísticas, tecnológicas e em toda e qualquer frente onde interações entre interlocutores se fazem necessárias.

Uma simples conversação entre duas pessoas contém um vasto leque de informações que transcendem os campos linguísticos: mesmo que o contato visual não se faça presente, a escuta e a análise de relações comunicativas permitem contruir, de forma indireta, identidades para os protagonistas. Características como gênero, idade, altura, porte, humor e afetividade são inferidas pelos ouvintes durante o processo, favorecendo assim a criação de uma imagem subjetiva dos interlocutores (1).

Neste cenário, a voz é responsável por carregar informações de identidade extremamente relevantes. Muito além da infererência sobre características físicas e pessoais dos locutores, elementos sonoros que vão do timbre da voz até padrões de pronúncia relacionadas a sotaques regionais compõem um pacote de conhecimento que, consumido pelos ouvintes, proporciona insumos suficientes para reconhecer e identificar o gerador de tais informações. De fato, mesmo após longos períodos, o ser humano é capaz de utilizar a voz para o reconhecimento de pessoas com um alto nível de acurácia (2).

Ao longo dos anos, diversos estudos foram realizados com o objetivo de propor um melhor entendimento sobre sinais sonoros, da sua geração até a sua percepção. Como resultado, uma série de aplicações puderam ser desenvolvidas nos mais diversos ramos de processamento de sinais, abrangendo tópicos vinculados ao monitoramento de eventos e atividades (3, 4), modelos de classificação de humor e estado afetivo (5), transcrições automáticas de voz em texto (6), reconhecimento e identificação de pessoas através da fala (7, 8), entre outros.

Em uma visão geral, com o crescente poder computacional gerado por unidades de processamento gráfico (GPUs, do inglês *Graphic Processing Units*) (9), o advento de algoritmos otimizados capazes de realizar cálculos em espaços multidimensionais (10) e, por fim, o aprimoramento de *frameworks* e bibliotecas voltados especificamente para o tratamento de sinais (11), aplicações cujo objetivo seja descrever, analisar ou interpretar áudios de modo a construir modelos inteligentes, se mostram altamente promissoras nos dias atuais.

Assim, nas próximas seções deste trabalho de graduação, será proposta uma imersão no entendimento de algumas das principais características de áudio extraídas de sinais de

vozes humanas com o objetivo de construir modelos de aprendizado de máquina capazes de identificar locutores a partir da fala.

## 1.1 Reconhecimento Automático de Locutores através da Fala

Em um ambiente pessoal, como a sala de estar de uma residência, diversos elementos tecnológicos atuam, individual ou coletivamente, de modo a proporcionar experiências personalizadas a seus usuários. Assistentes pessoais, TVs, *smartphones*, consoles e muitos outros aparelhos utilizam, como gatilho, demandas de seus portadores para realizar as mais variadas ações: tocar uma música, acessar páginas de aplicativos, iniciar um gerenciador de bate papo, ajustar a temperatura do ar condicionado ou até mesmo modificar o conjunto de luzes de um determinado cômodo.

Neste contexto, por ser considerada uma das formas mais naturais de comunicação humana, a voz é utilizada por muitas empresas como gatilho acionador desses aparelhos inteligentes. Estudos revelam que, até 2030, milhões de pessoas utilizarão a voz para interagir com máquinas em um cenário onde dispositivos orientados a comandos serão ferramentas comuns presentes em um leque de aplicações que abrange desde aparelhos celulares até paineis internos de automóveis (12). Em razão disso, a verificação e a identificação de locutores se tornam essenciais em meio ao grande leque de possibilidades do mundo moderno.

Em uma definição técnica, dado um segmento de fala $Y$ e um locutor hipotético $S$, a tarefa relacionada à identificação, também associada à verificação de locutores, envolve determinar, de forma direta, se $Y$ foi pronunciado por $S$, assumindo que $Y$ contém elementos de fala oriundos de apenas um único locutor (13).

Como pode ser visto na Figura 1, as etapas de construção de um modelo capaz de reconhecer locutores envolvem, de maneira geral, a conversão de um sinal analógico em uma representação digital capaz de ser computacionalmente processada (detalhes sobre esse processo serão fornecidos na seção 2.1). Já a extração de características de um sinal de áudio tem como objetivo a caracterização do sinal em grandezas matemáticas, permitindo assim criar uma identidade computacional para o referido áudio (definições sobre este tópico serão fornecidas a partir da seção 2.2). Por fim, a construção de uma base de dados é um passo essencial para o treinamento de modelos de aprendizado capazes de utilizar os elementos matemáticos extraídos na etapa anterior para, de fato, construir regras de decisão de modo a retornar respostas atreladas à probabilidade de um dado sinal de entrada pertencer a um determinado locutor presente na base. No âmbito do consumo desse sistema, sinais de áudio de locutores desconhecidos passam pelo processo de caracterização até que os atributos extraídos possam ser validados por um modelo já treinado e com a capacidade de retornar uma resposta válida de acordo com o objetivo do

sistema.

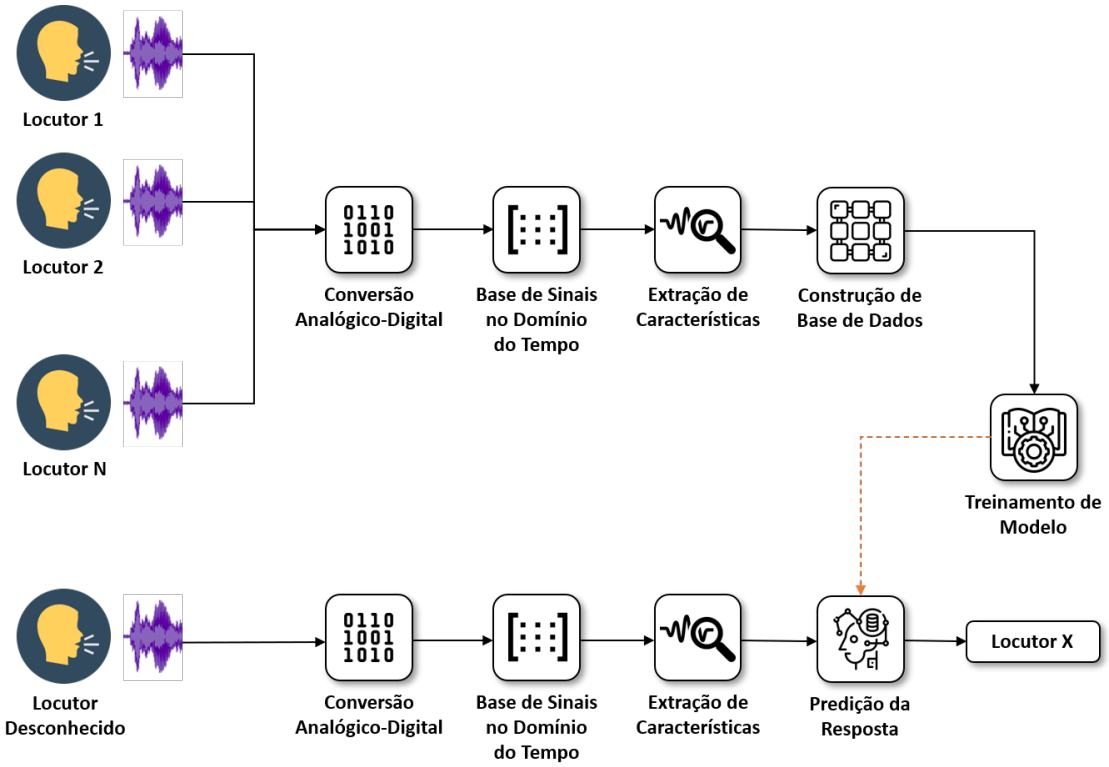


Figura 1 – Exemplo de diagrama de construção e utilização de um modelo de identificação de locutores a partir da extração de características de sinais digitais gerados a partir de diferentes usuários.

Em (7, 14), o universo relacionado ao reconhecimento de locutores através da fala é definido em duas principais frentes: identificação e verificação. Em processos de identificação, a principal tarefa é identificar um sinal de entrada em um conjunto de locutores definido previamente. Quando, nessa ação, são utilizadas apenas entradas de áudio de locutores conhecidos, é dito se tratar de um problema de *in-set scenario*. Por outro lado, quando a tarefa de identificação é realizada através de sinais de locutores externos à lista predefinida, trata-se de um problema denominado como *out-set scenario*, exigindo assim a utilização de uma base geral e universal capaz de proporcionar uma resposta externa aos locutores conhecidos na base. Já a frente de verificação diz respeito a um locutor desconhecido reivindicando sua identidade através da fala em uma tarefa que tem como principal objetivo validar se essa requisição é verdadeira.

Entre as aplicações, ainda é possível analisar o reconhecimento de fala considerando a dependência ou a independência do texto pronunciado. Em outras palavras, modelos dependentes de texto exigem, além da autenticação relacionada ao locutor, a validação do texto pronunciado pelo mesmo. Analogamente, modelos independentes de texto não requerem a pronúncia específica de uma determinada frase ou palavra para o processo de autenticação.

Como mencionado anteriormente, seres humanos possuem a incrível habilidade de reconhecer locutores através da fala mesmo expostos a poucas situações de contato com os mesmos. Especialistas forenses e peritos em áudio possuem habilidades ainda mais refinadas para essa tarefa, exercendo um papel fundamental na resolução de casos criminais através de análises detalhadas em áudios muitas vezes gravados em situações extremas. Entretanto, considerando a abordagem automática do reconhecimento de fala, é preciso utilizar técnicas computacionais de modelagem para construir um sistema capaz de receber sinais de áudio e retornar a saída esperada sem a interação humana, seja esta relacionada à identificação ou à verificação de locutores.

## 1.2 A Ideia por Trás da Modelagem do Áudio

Entre a coleta de amostras de áudio de locutores e a criação de um modelo capaz de aplicar o reconhecimento adequado, existe um grande caminho a ser percorrido. Considerando a atuação humana nesse ofício, é relativamente complexo definir, ao certo, quais parâmetros ou critérios são utilizados na validação ao ouvir a voz de alguém conhecido. Em uma experiência prática, é possível pontuar termos como “presença”, “brilho” e “cor” o que, muitas vezes, dificulta o entendimento técnico dos atributos de áudio associados. Computacionalmente, o primeiro grande desafio na construção de modelos de reconhecimento de locutores a partir da fala é, de fato, a extração de características relevantes destes sinais, capazes de serem matematicamente interpretáveis. Tais características são popularmente conhecidas como parâmetros ou *features*.

De forma geral, o principal papel de uma *feature* é proporcionar uma boa descrição ou diferenciação de um determinado objeto alvo do estudo. Assim, é dito que uma *feature* é relevante quando esta é capaz de providenciar características expressivas do objeto alvo em relação a um contexto. No universo relacionado ao reconhecimento de locutores através da fala, tais *features* podem ser definidas como características extraídas dos sinais de áudio gravados dos interlocutores após a conversão analógico-digital. Conforme pontuado em (7, 15), a definição de um conjunto ideal de parâmetros a serem extraídos de sinais de áudio envolve:

- Alta variação entre diferentes locutores e baixa variação para um mesmo locutor
- Alta robustez contra ruídos e distorções
- Frequência e naturalidade no processo de fala
- Fácil extração e medição a partir do sinal de fala
- Dificuldade de disfarce ou imitação por possíveis impostores
- Inerência ao estado de saúde do locutor ou a variações temporais

As propriedades mencionadas são essenciais na construção de modelos eficientes de

reconhecimento de fala. Ao longo deste trabalho, algumas das principais características comumente extráidas em aplicações semelhantes serão detalhadas de modo a propor um entendimento claro sobre os sinais de áudio.

## 1.3 Objetivo

Este trabalho de graduação tem por objetivo a construção de um modelo de reconhecimento de locutores através da fala considerando uma abordagem *out-set scenario* independente de texto. Para isso, será proposta a criação de uma base de dados a partir de gravação de sinais de áudio referentes a locutores predefinidos. Posteriormente, essa base será utilizada no tratamento, processamento e extração de *features* de áudio nos domínios do tempo, frequência e tempo-frequência de modo a alimentar modelos de Aprendizado de Máquina em um contexto de aprendizado supervisionado (classificação multiclasse).

Considerando a utilização do modelo por qualquer pessoa, será utilizada uma base pública de gravações de áudios em português disponível no portal Mozilla Common Voice (16) para generalizar interlocutores que não tiveram gravações registradas e, portanto, não fizeram parte de uma classe específica do modelo durante as etapas de treinamento.

# 2 Fundamentação Teórica

## 2.1 Análise Digital de Sinais de Áudio

Dentro das diretrizes relacionadas ao presente trabalho de graduação, faz-se necessário definir os blocos fundamentais associados a ondas sonoras em um contexto específico de sinais de áudio sob uma ótica digital. Com isso, será possível propor estudos mais refinados sobre as possibilidades de tratamento e processamento de sinais de fala obtidos naturalmente em meios comuns.

Em (17), é dito que sinais analógicos estão conectados a quantidades físicas, como comprimento, peso, volume ou, mais apropriadamente no campo de eletrônicos: tensão, carga, corrente e fluxo magnético. Em algumas aplicações práticas, o processamento de sinais é de tal modo simples que circuitos puramente analógicos (amplificação de áudio e filtragem) atendem perfeitamente as necessidades. Entretanto, com o advento de problemas mais complexos a serem resolvidos, a análise de sinais em ambientes digitais surgiu como uma importante frente capaz de proporcionar, entre diversos fatores, uma melhor lógica de armazenamento, possibilidade de cálculos complexos e uma alta capacidade de adaptação de algoritmos. Se uma aplicação necessita de uma dessas vantagens, então sinais analógicos devem ser convertidos para sinais digitais no início do fluxo de processamento.

De maneira geral, sinais sonoros representados pela voz humana podem ser entendidos como formas de onda geradas a partir do distúrbio de pressão das moléculas de ar oriundo da fala. Caracterizados como ondas mecânicas no meio analógico, tais sinais necessitam de uma conversão analógico-digital para serem tratados e analisados computacionalmente, permitindo assim com que os elementos contínuos de amplitude e tempo sejam transformados em blocos discretos tratados em linguagem computacional.

Ao longo dos próximos tópicos dessa seção, as técnicas de conversão e processamento e sinais serão abordadas em maiores detalhes para que, ao longo dessa fundamentação teórica, seja possível compreender as etapas de caracterização de modelagem a serem detalhadas posteriormente.

### 2.1.1 Amostragem

O processo de amostragem de um sinal analógico refere-se à discretização deste sinal no tempo a partir da retirada de amostras em intervalos específicos (18). De acordo com (19), existem três estágios a serem realizados durante o processo de amostragem:

1. É definida uma frequência de amostragem para coleta periódica de amostras da

amplitude do sinal

2. As amostras coletadas representam o valor da amplitude do sinal em um determinado instante de tempo
3. O valor de amplitude amostrado pode ser utilizada em etapas posteriores de quantização

Em (18), é dito que o período de amostragem $T_s$, medido em segundos, entre pontos de amostragem é equidistante no contexto de sinais de áudio e dado por:

$$T_s = \frac{1}{f_s} \tag{2.1}$$

onde $f_s$ é o termo definido por *frequência de amostragem* de um sinal. A Figura 2 exemplifica um processo de amostragem em um sinal analógico (parte superior) e sua representação em tempo discreto (parte inferior). Independente dos valores associados à frequência do sinal e a frequência de amostragem, a representação do sinal em tempo discreto é obtida através da coleta de um número predefinido de amostras a cada segundo. Em outra notação comumente utilizada, $f_s$ também pode ser encontrada como $s_r$ (do inglês, *sample rate*).

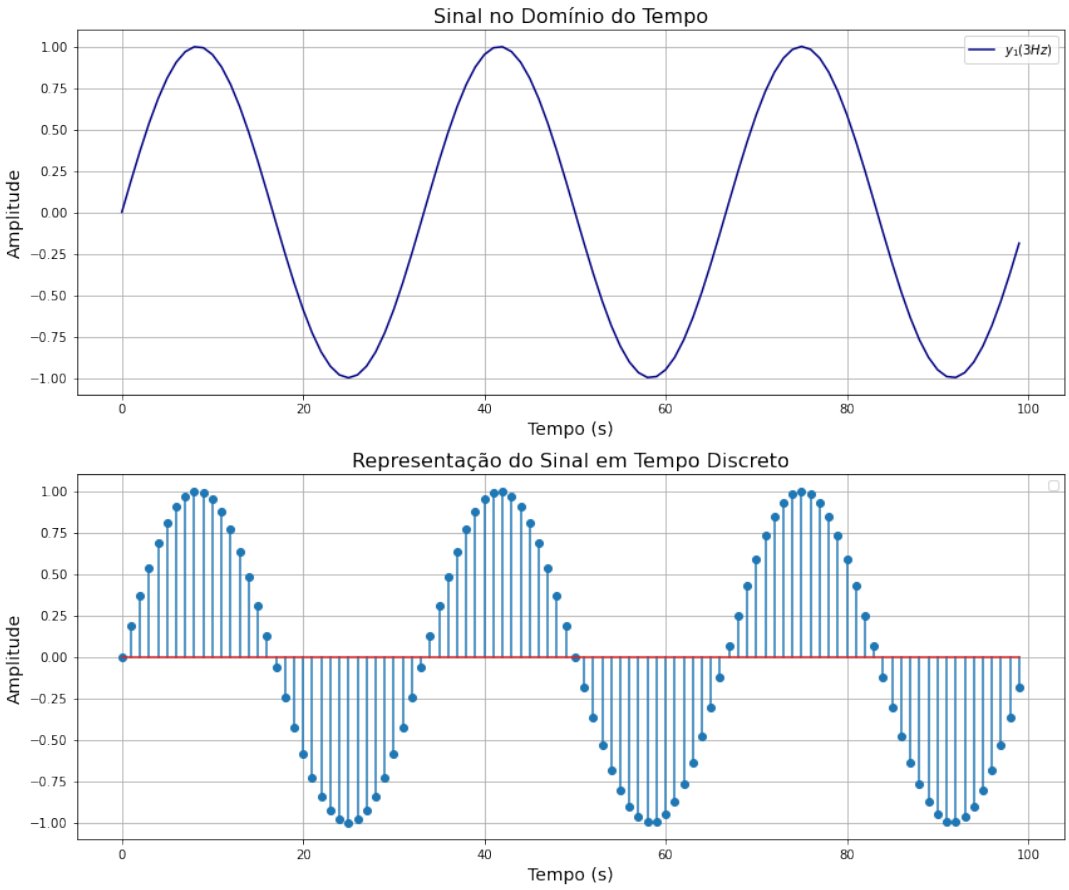


Figura 2 – Sinais contínuo e amostrado a partir de uma dada taxa de amostragem.

Uma importante propriedade de sinais amostrados é a possibilidade de ambiguidade em suas representações digitais. Na Figura 3, é possível visualizar diferentes sinais analógicos de entrada gerando sinais amostrados idênticos (18).

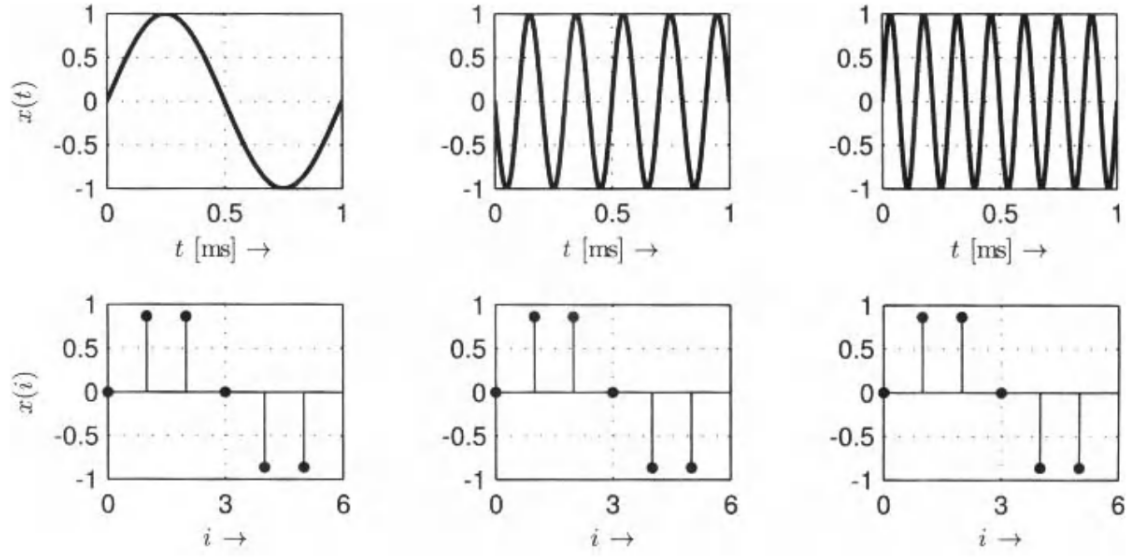


Figura 3 – Diferentes sinais contínuos (parte superior) e amostrados (parte inferior) com diferentes frequências (da esquerda para a direita: 1 kHz, 5 kHz e 7 kHz). A frequência de amostragem é $f_s = 6$ kHz.

Analisando os sinais amostrados presentes na figura 3, seria impossível alcançar as respectivas representações analógicas de forma correta. Assim, é possível dizer que um sinal só pode ser reconstruído por completo quando alguns requerimentos forem cumpridos. Neste cenário, define-se então o **teorema de amostragem**, afirmando que um sinal amostrado só pode ser reconstruído, sem perdas de informação, se a taxa de amostragem $f_s$ for maior que duas vezes a frequência máxima $f_{max}$ do sinal de áudio amostrado (18). A esta relação dá-se o nome de *critério de Nyquist*, definindo assim a frequência ótima de amostragem capaz de reconstruir o sinal sem perdas por:

$$f_s >= 2 \cdot f_{max} \tag{2.2}$$

A faixa audível do ouvido humano é entre 20 e 20000 Hz aproximadamente e, portanto, elementos sonoros fora desse intervalo podem ser considerados imperceptíveis. Analisando os critérios estabelecidos, para construir uma aplicação de tratamento digital de sinais envolvendo interações humanas em uma taxa de amostragem sem que haja perdas perceptíveis, é possível levar em consideração o Teorema de Amostragem de Nyquist-Shannon.

Em diversas aplicações, são encontradas taxas de amostragens de 44100 Hz ou 22050 Hz, sendo o primeiro caso muito popular em análises musicais sem nenhuma perda de frequência perceptível, visto que $f_N = \frac{f_s}{2} = 22050$ Hz supera o limiar humano. Em

estudos relacionados a reconhecimento de voz, taxas de amostragem de 22050 Hz são popularmente usadas por reservar espaço sem perdas significantes. Em (20), é possível encontrar motivações para o uso de diferentes taxas de amostragem.

Após amostrar um sinal de áudio contínuo, faz-se necessária a aplicação de processos capazes de transformar os valores resultantes do processo de amostragem em uma codificação capaz de ser processada computacionalmente. A seguir, o processo de quantização será detalhado com essa finalidade.

### 2.1.2 Quantização

A quantização, de maneira direta, tem o papel de transformar ou arredondar os valores amostrados no tempo, dispostos entre dois limites, para valores quantizados. Durante a quantização, o nível do sinal no momento da amostragem é comparado com um número limitado de valores analógicos em uma ideia de discretização de amplitude (17). Em outras palavras, na quantização, cada valor de amplitude do sinal de entrada é arredondado considerando um conjunto predefinido de valores de amplitudes (18).

A Figura 4 mostra um exemplo de quantização considerando um conversor analógico digital de um sinal de entrada. Nela, é ilustrado um elemento de tensão analógica de entrada $V_{in}$ em um conversor analógico digital que, disposto de uma dada frequência de amostragem $f_s$ e um número finito de $N$ bits, realiza a discretização deste sinal dentro de uma faixa restrita de valores.

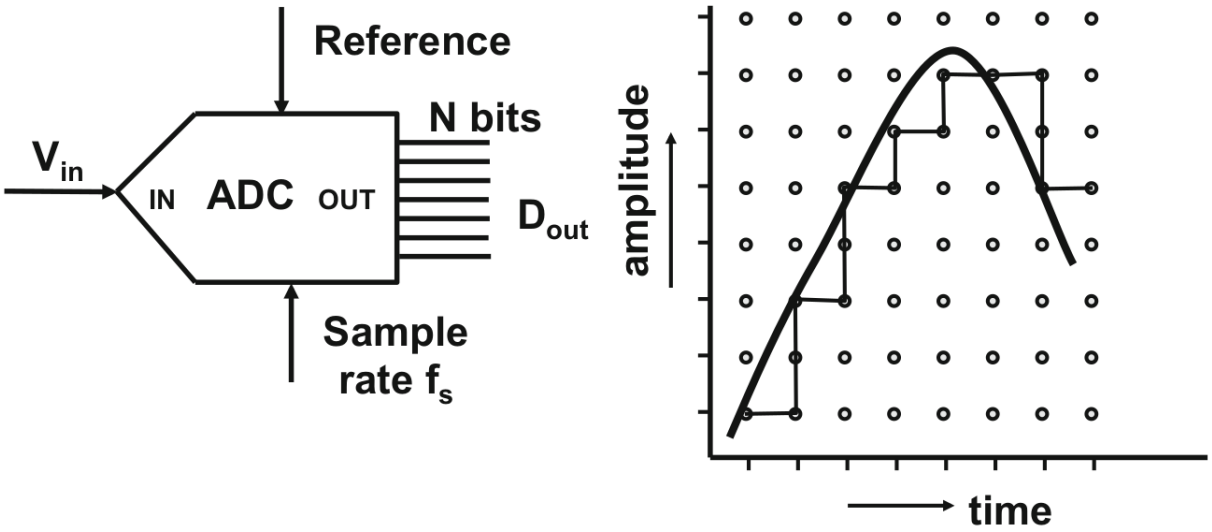


Figura 4 – Um conversor analógico-digital arredonda a amplitude de um sinal contínuo, no momento da amostragem, para o valor mais próximo da escala discreta de amplitude definida.

Grande parte das representações discretas de amplitude utiliza bases binárias e,

dessa forma, um número total de bits deve ser definido para formar um sinal digital. Nesse contexto, a esse número dá-se o nome de *tamanho da palavra* ou *resolução* representado pelo termo $N$. Assim, um sinal digital quantizado está limitado a $2^N$ possíveis valores ou níveis que definem a amplitude discretizada (17). Em (18), é dito que resoluções típicas atreladas a sinais de áudio são 16, 24 e 32 bits.

Analisando a essência da quantização, é possível observar imperfeições ou possíveis lacunas características de qualquer processo de digitalização. Para avaliar a qualidade da quantização, é possível observar a relação sinal-ruído (ou SNR), sendo esta dada por:

$$\text{SNR} = 10 \log_{10} \left( \frac{P_s}{P_q} \right) \text{ [dB]} \tag{2.3}$$

onde $P_s$ indica a potência do sinal e $P_q$ indica a potência do erro de quantização.

### 2.1.3 Enquadramento

Grande parte dos procedimentos de extração de características em sinais digitais de áudio utilizam blocos ou conjuntos de amostras extraídos do sinal de entrada (18) e, a esse procedimento, dá-se o nome de enquadramento (do inglês, *framing*). Em geral, para aplicar o devido enquadramento em um sinal de áudio qualquer, define-se um parâmetro associado a quantidade de amostras a ser contemplada em cada bloco (ou *frame*), permitindo assim a divisão de um sinal completo em pacotes menores de amostras. A Figura 5 ilustra um procedimento teórico de *framing* em um sinal de áudio onde cada retângulo vermelho representa um *frame*.

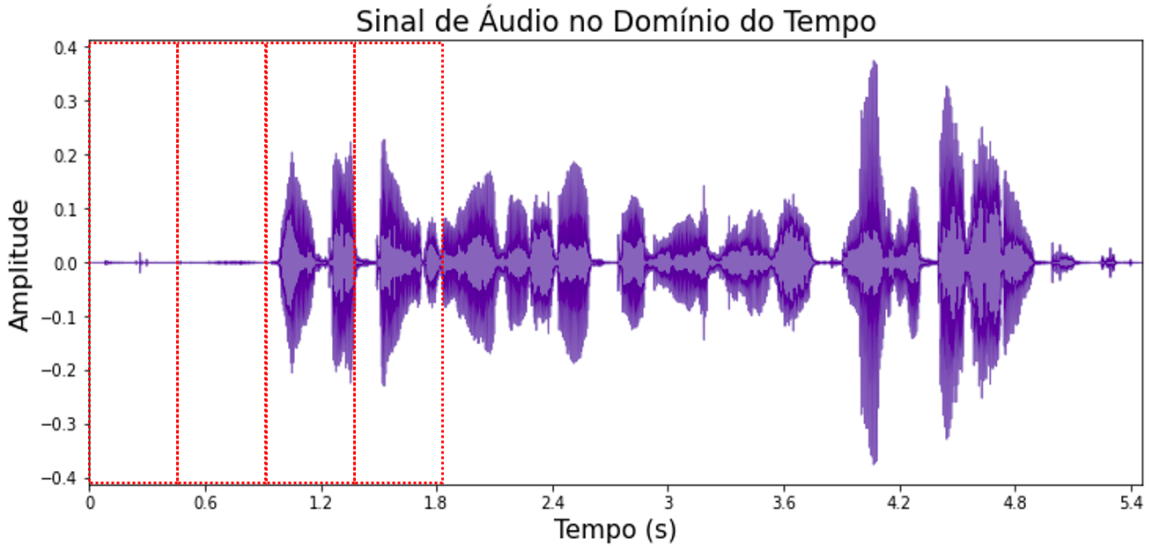


Figura 5 – Separação de um sinal de áudio em diferentes blocos de processamento conhecidos como *frames*.

Considerando o parâmetro $s_r$ como a taxa de amostragem de um sinal de áudio,

ou seja, a quantidade de amostras presentes em cada segundo, o intervalo de tempo entre amostras de um sinal amostrado a uma taxa $s_r = 22000$ Hz é dado por:

$$d_s = 1 \text{ amostra} \times \frac{1}{22000 \text{ amostras / s}} = 0,0000454 \text{ s} = 0,0454 \text{ ms}$$

Assim, considerando um *frame* com um total de $K$ amostras, é possível definir a equação que retorna a duração de cada *frame* como:

$$d_f = K \cdot \frac{1}{s_r} \tag{2.4}$$

Para o parâmetro $K$, normalmente é escolhido um número proporcional à potência de 2 de modo a agilizar cálculos computacionais. Assim, em aplicações práticas, é possível encontrar valores de $K$ variando entre 256 e 8192.

### 2.1.4 Janelamento

Ao aplicar o processo de separação de um sinal digital de áudio em diferentes *frames*, nota-se o surgimento de descontinuidades no sinal presentes em cada bloco individual. Para este entendimento, é possível tomar a Figura 5 como base e observar os valores exatos de amplitude da última amostra de um dado *frame* e da primeira amostra do *frame* imediatamente posterior. Na prática, haverá distinção entre esses valores de amplitude, ocasionando assim descontinuidades entre os diferentes sinais enquadrados. Assim, o processo de janelamento (do inglês, *windowing*) consiste na aplicação de uma função de janela em cada *frame* do sinal para remoção de tais descontinuidades (21, 22).

Na prática, uma escolha comum relacionada a função de janela é dada pela função Hann, definida por:

$$w(k) = 0.5 \cdot \left(1 - \cos\left(\frac{2.\pi.k}{K-1}\right)\right) \rightarrow k = 1...K \tag{2.5}$$

Graficamente, a Figura 6 mostra um exemplo da função Hann para 50 amostras. Já a Figura 7 ilustra um procedimento completo de janelamento aplicado a um único *frame* extraído de um sinal de áudio contendo 256 amostras. Ao multiplicar cada *frame* com o respectivo valor da função Hann dada pela equação 2.5, eventualmente evita-se o que é conhecido como "efeito de borda" que, de fato, pode ocasionar um problema comumente encontrado no tratamento de sinais no domínio da frequência: o vazamento de frequências (ou *spectral leakage*). Na prática, o vazamento de frequências é um fenômeno gerado pela aplicação da Transformada de Fourier em sinais não periódicos, ocasionando assim componentes indesejadas de altas frequências no espectro resultante.

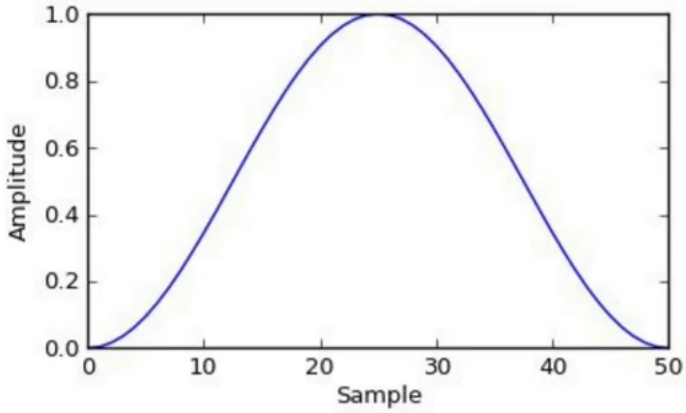


Figura 6 – Visualização gráfica da função Hann para $K = 50$ amostras.

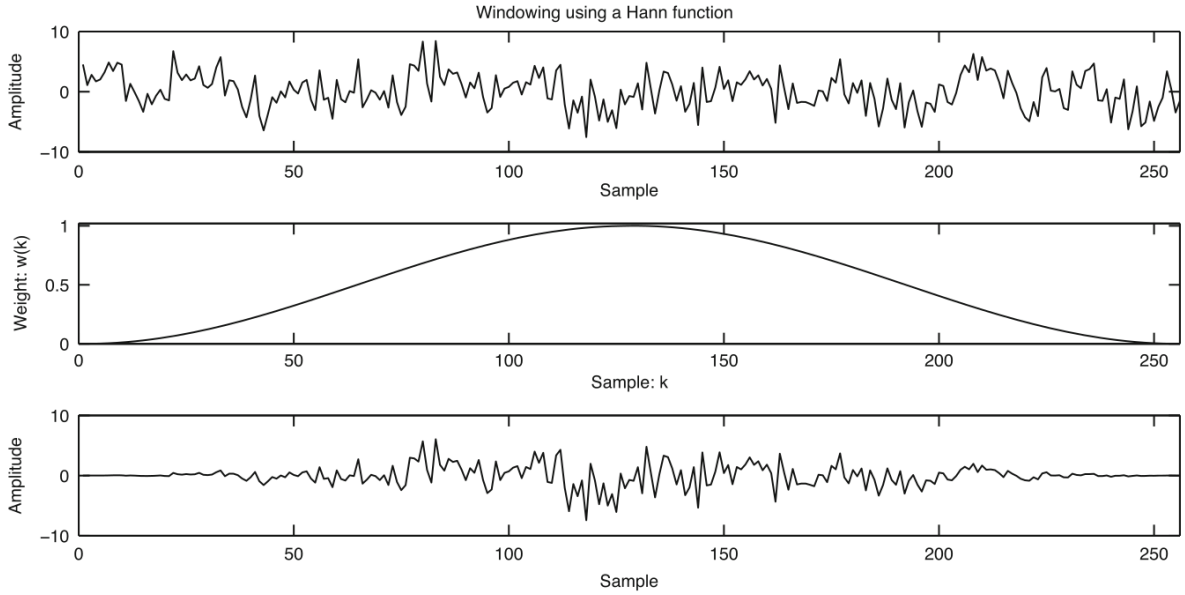


Figura 7 – Aplicação do processo de janelamento através da função Hann.

A equação do sinal resultante após o procedimento de janelamento é dada por:

$$s_w(k) = s(k) \cdot w(k) \tag{2.6}$$

onde $s(k)$ representa o sinal de áudio amostrado e $w(k)$ representa a função *window* aplicada.

### 2.1.5 Sobreposição de Frames e Hop Size

Como pôde ser visto no processo de janelamento, com o objetivo de evitar o "efeito de borda", a aplicação de uma função *window* se faz necessária para a devida suavização dos pontos de início e fim do sinal, fazendo assim com que a amplitude alcance o nível zero nesses dois cortes. Entretanto, na prática, essa suavização ocasiona perdas de informação,

visto que os níveis lógico do sinal nessas pontas foi migrado essencialmente para zero, modificando diretamente os níveis presentes no sinal original e não-periódico do *frame*.

De modo a evitar essa perda de informação ocasionada pela supressão, normalmente aplica-se um processo de sobreposição de *frames* (ou *overlapping*) em uma certa quantidade definida pelo parâmetro *hop size* (21). Na Figura 5, introduzida na sessão de enquadramento, foi possível visualizar uma proposta teórica e ilustrativa de separação de um sinal de áudio em diferentes blocos de amostras. Considerando então a sobreposição desses blocos de modo a evitar perdas de informação, a Figura 8 traz consigo uma proposta de enquadramento juntamente com o parâmetro *hop size* definindo o nível de sobreposição de cada quadro.

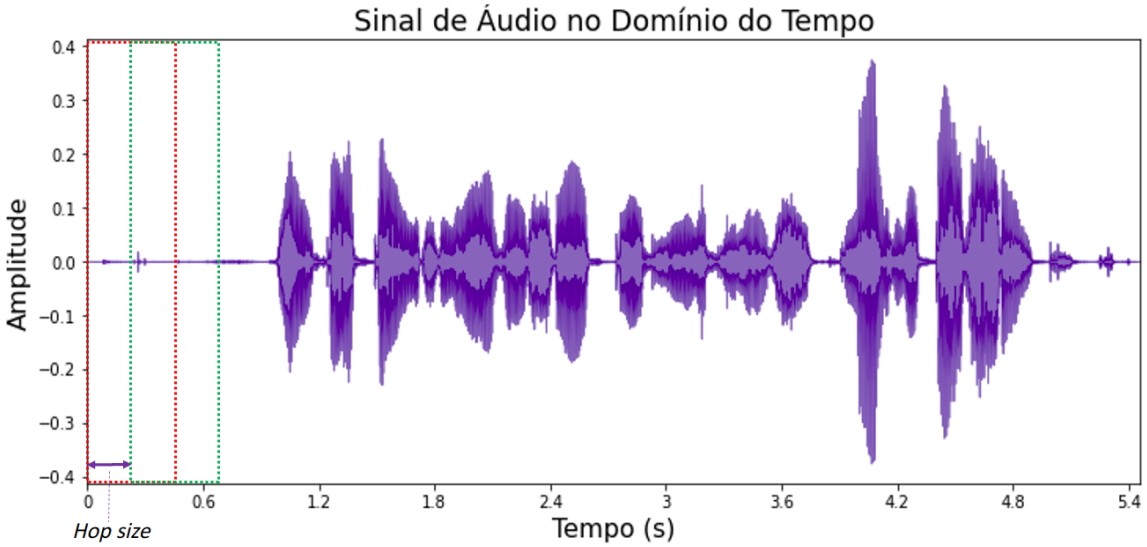


Figura 8 – Processo de sobreposição de *frames* através do parâmetro *hop size*.

Em geral, o parâmetro *hop size* é selecionado de tal forma a proporcionar uma sobreposição de aproximdamente 50% nos *frames* associados ao sinal. Na prática, considerando um tamanho de frame $K = 256$ amostras, é comum observar aplicações com uma sobreposição associada a 128 amostras.

Assim, dadas as definições gerais dos processos de enquadramento, janelamento e sobreposição de sinais, é possível iniciar o detalhamento das características de modo a extrair significado computacional dos mesmos. Nas próximas seções, os conceitos definidos até aqui serão utilizados como blocos fundamentais dentro das tratativas relacionadas à extração de atributos dos sinais de áudio em diferentes domínios.

## 2.2 Atributos de Sinais de Áudio no Domínio do Tempo

Após uma visão geral sobre os principais blocos associados à transcrição de um sinal analógico para o domínio de digital, além dos procedimentos normalmente adotados para que um sinal digital de áudio possa ser devidamente caracterizado, é chegado o momento de definir, de fato, formas de extração de atributos destes sinais de modo a propor análises

mais complexas de modelagem. Neste tópico, serão descritas as características de áudio quando estes se encontram no domínio do tempo, ou seja, em sua forma mais conhecida até o momento. Neste cenário, tem-se em mãos uma série temporal de elementos associados à amplitude dos sinais de áudio lidos e digitalizados.

### 2.2.1 Envelope de Amplitude

Basicamente, o envelope de amplitude pode ser retornado através do valor máximo de amplitude considerando todas as $K$ amostras de um frame $t$ (21). Sua definição formal é dada por:

$$AE_t = \max_{k=t \cdot K} s(k) \tag{2.7}$$

Como características principais, é possível afirmar que o envelope de amplitude é um atributo sensível a *outliers* e aplicado em processos de identificação de início de notas musicais (21).

### 2.2.2 Raíz da Energia Média Quadrática

A raíz da energia média quadrática (também denotada como energia RMS, do inglês, *Root Mean Squared Energy*) é calculada a partir da energia extraída de cada frame de um sinal de áudio no domínio do tempo. Em (21), é dito que a energia RMS pode ser utilizada como um boa estimativa de intensidade do som e também um indicador para identificação de novos eventos em aplicações de segmentação de áudio. Menos sensível a *outliers* do que o envelope de amplitude, o cálculo da energia RMS para cada frame $t$ é dado por:

$$RMS_t = \sqrt{\frac{1}{K} \sum_{k=t \cdot K}^{(t+1)K-1} s^2(k)} \tag{2.8}$$

### 2.2.3 Zero Crossing rate

Intuitivamente, o atributo *zero crossing rate* (ou ZCR) mede o número de vezes que o valor de amplitude muda de sinal considerando cada frame $t$ (21). Seu cálculo é dado por:

$$ZCR_t = \frac{1}{2} \sum_{k-tK}^{(t+1)K-1} |sgn(s(k)) - sgn(s(k+1))| \tag{2.9}$$

onde a função $sgn(s(k))$ refere-se à função sinal[1] da amostra $k$ dentro do frame $t$. Os termos $sgn(s(k)) - sgn(s(k+1))$ indicam uma comparação entre o sinal da amostra $k$ e o sinal da amostra $k+1$ subsequente, trazendo assim a ideia de verificação se o sinal realmente cruzou o eixo zero horizontal.

Como aplicações práticas, o *zero crossing rate* pode ser utilizado em aplicações de reconhecimento de fala, decisões relacionadas à voz e silêncio em sinais de fala e em classificação musical.

## 2.3 Atributos de Sinais de Áudio no Domínio da Frequência

Nesta sessão, serão abordados tópicos relacionados à extração de atributos de sinais de áudio no domínio da frequência. Até o presente momento, as características detalhadas na fundamentação envolveram, como entrada, o sinal puro no domínio do tempo. Neste cenário, é preciso definir um conceito essencial vinculado à transformação deste sinal temporal em um espectro de frequências: a Transformada de Fourier.

### 2.3.1 Transformada de Fourier

De forma direta, a Transformada de Fourier é uma ferramenta clássica que permite obter a representação de um sinal no domínio da frequência.

Em um caráter intuitivo, como uma forma de entender o conceito básico da análise de Fourier em um aspecto harmônico, em (19) é possível observar que, se um sinal de áudio passa sobre um filtro passa-faixa, de ganho unitário, bem estreito, capaz de varrer todo o sinal, a saída deste filtro indicará a magnitude de cada banda de frequências que representa o dado sinal. A largura dessa banda determina o quão correto será o conteúdo dessa análise de frequências.

Para sinais contínuos, a Transformada de Fourier é dada por:

$$\mathcal{G}(f) = \mathcal{F}\{g(t)\} = \int_{-\infty}^{+\infty} g(t) \cdot e^{-i2\pi ft} dt \tag{2.10}$$

Entretanto, em termos de análises digitais em sinais, é utilizada a Transformada Discreta de Fourier (ou DFT, do inglês *Discrete Fourier Transform*), definida por:

$$X_m = \sum_{k=0}^{K-1} x_k \cdot e^{-i2\pi \frac{k.m}{K}} \tag{2.11}$$

onde $m$ é um número inteiro variando entre 0 e $K-1$, $K$ representa o número de amostras em cada *frame* e $x_k$ representa o k-ésimo elemento de amplitude do sinal de entrada, ou

[1] Definição da função sinal $sgn(x)$: 1 se $x > 0$, 0 se $x = 0$ e $-1$ se $x < 0$

seja, a k-ésima amostra no *frame*. Cada elemento de $X_m$ é um número complexo contendo uma parte real $Re(X_m)$ e uma parte imaginária $Im(X_m)$, das quais podem ser obtidas a magnitude a partir da equação (2.12) e a fase a partir da equação (2.13) como em (21):

$$|X_m| = \sqrt{Re^2(X_m) + Im^2(X_m)} \tag{2.12}$$

$$\phi(X_m) = \tan^{-1} \frac{Im(X_m)}{Re(X_m)} \tag{2.13}$$

Um algoritmo computacionalmente mais eficiente da Transformada de Fourier é dada pela Transformada Rápida de Fourier (FFT, do inglês *Fast Fourier Transform*), a qual é amplamente utilizada em *frameworks* e bibliotecas computacionais para análises relacionadas. Como exigência para a utilização prática da FFT, define-se um número $K$ de amostras proporcional a uma potência de 2, garantindo assim sua eficiência.

Considerando a definição formal da Transformada de Fourier, é possível compreender a decomposição de um sinal amostrado de entrada, originalmente dado como uma série temporal de amplitudes, em diferentes frequências e suas respectivas magnitudes, gerando assim um espectro único capaz de proporcionar uma análise detalhada sobre as frequências que compõe o dado sinal. Com isso, um novo leque de características a serem extraídas dos sinais se abre em um contexto fixo de análise espectral sem a dimensão temporal.

A análise do espectro resultante da DFT não proporciona uma visão dinâmica sobre o momento exato em que as dadas frequências do espectro são perceptíveis, mas sim uma única foto das magnitudes de cada componente. Em meio a este cenário, é possível definir a Transformada de Fourier de Tempo-Curto (STFT, do inglês *Short Time Fourier Transform*) como uma transformada capaz de contemplar o aspecto temporal do espectro de frequências. Em (21) pontua-se que a STFT basicamente calcula uma série de DFTs ao longo de diferentes segmentos (*frames*) de um sinal de entrada, resultando assim em uma matriz capaz de retornar análises específicas como espectrogramas.

Matematicamente, a STFT é dada por:

$$S(m,k) = \sum_{n=0}^{K-1} x(n+mH) \cdot w(n) \cdot e^{-i2\pi n \frac{k}{K}} \tag{2.14}$$

onde os termos são definidos por:

- $m$: índice do frame relacionado ao sinal original
- $H$: parâmetro *hop size* (o termo $mH$ representa a primeira amostra do frame)
- $w(n)$: função *window* (Hann)

Como resultado final, a STFT retorna uma matriz espectral de duas dimensões que relaciona tempo e frequência em um mesmo objeto. Na prática, essa matriz é composta por um eixo y relacionado a cada uma das frequências resultantes da transformada de Fourier e por um eixo x representando cada um dos *frames* extraídos durante a análise. Cada elemento dessa matriz traz os coeficientes complexos da transformada de Fourier para cada faixa de frequência em cada *frame* (ou em cada espaço de tempo do sinal).

As dimensões da matriz resultante (#frequências, #frames) podem ser calculadas por:

$$\#\text{ frequências} = \frac{framesize}{2} + 1 \;\rightarrow \text{entre 0 e } s_r/2$$

$$\#\text{ frames} = \frac{samples - framesize}{hopsize} + 1$$

Pela natureza do cálculo da STFT, surge uma relação entre as dimensões de tempo e frequência resultantes. Quanto maior o tamanho $K$ do *frame*, maior a resolução da frequência e menor a resolução do tempo, visto que o sinal será dividido em um número menor de blocos. Analogamente, quanto menor o valor para $K$, menor o número de frequências analisadas na transformada e maior o número de *frames* considerados no eixo temporal.

Uma vez apresentado como obter a representação espectral de um sinal, serão descritas, nas próximas seções, as características espectrais a serem extraídas dos sinais de áudio.

### 2.3.2 Taxa de Energia de Banda

De acordo com (21), a taxa de energia de um sinal (ou BER, do inglês *Band Energy Ratio*) é um atributo computado no domínio da frequência e utilizado comumente para discriminação entre elementos de fala e elementos musicais em sinais de áudio. Em conjunto com outras características, pode também ser aplicada para a classificação de gênero musical. Seu cálculo relaciona a energia em bandas de baixa frequência com a energia em bandas de alta frequência em uma ideia de transcrição sobre o quão dominante são as baixas frequências. Sua fórmula é dada por:

$$BER_t = \frac{\sum_{n=1}^{F-1} m_t(n)^2}{\sum_{n=F}^{N} m_t(n)^2} \tag{2.15}$$

onde os termos podem ser definidos por:

- $F$: parâmetro que delimita a frequência separadora de banda, ou seja, o valor de frequência que define altas e baixas frequências

- $m_t(n)$: termo referente à magnitude do sinal no domínio da frequência no *frame* $t$ e banda de frequência $n$

Em aplicações relacionadas a tratamento de sinais de voz, normalmente utiliza-se um valor de 2000 Hz para a frequência $F$ de corte e este valor está intimamente ligado à grande concentração de informações relacionadas à fala contidas em frequências abaixo desse limiar.

### 2.3.3 Centroide Espectral

O centroide espectral de um sinal de áudio representa o centro de gravidade da magnitude do espectro de frequências, ou seja, a banda de frequência onde a maior parte da energia está concentrada. Este atributo é utilizado como uma medida de "brilho" do som, relacionado também ao timbre musical. Contudo, o centroide espectral é altamente sensível a filtros passa-baixa, uma vez que as bandas de alta frequência são dadas em um peso maior que as bandas de baixa frequência (21). A equação para seu cálculo é dada por:

$$SC_t = \frac{\sum_{n=1}^{N} m_t(n) \cdot n}{\sum_{n=1}^{N} m_t(n)} \tag{2.16}$$

onde, no numerador, é possível encontrar o fator $n$ relacionado à faixa de frequência ponderada por $m_t(n)$ (magnitude do sinal no *frame* $t$). No denominador, tem-se uma soma total dos pesos (amplitudes) do espectro todo dentro do *frame* $t$.

### 2.3.4 Largura de Banda

A largura de banda, também conhecida como espalhamento espectral, é uma característica derivada do centroide espectral e pode ser interpretada em uma ideia de variância da frequência média do sinal (21), visto que o grande objetivo da largura de banda é analisar o espalhamento de energia nos arredores do centroide espectral. Sua fórmula é dada por:

$$BW_t = \frac{\sum_{n=1}^{N} |n - SC_t| \cdot m_t(n)}{\sum_{n=1}^{N} m_t(n)} \tag{2.17}$$

onde o termo $|n - SC_t|$ representa a distância da banda de frequência $n$ do centroide espectral $SC_t$.

## 2.4 Atributos de Sinais de Áudio no Domínio Tempo-Frequência

Após analisar os atributos de sinais de áudio extraídos no domínio do tempo e no domínio da frequência, nesta sessão serão abordadas algumas das princípais características

de áudio no que pode ser chamado de domínio tempo-frequência. Nesse contexto, serão utilizados conceitos atrelados ao cálculo da STFT definida na seção 2.3.1 para propor análises onde se possa unir a dimensão temporal com a dimensão espectral.

Assim, visando enriquecer ainda mais o conteúdo referente aos atributos de sinais analisados neste trabalho de graduação, os próximos tópicos irão contemplar detalhes sobre características de sinais de áudio extraídas com base em características temporais (domínio do tempo) e espectrais (domínio da frequência).

### 2.4.1 Espectrogramas

Espectrogramas são visualizações típicas de sinais de áudio que combinam componentes de tempo e de frequência. A STFT é calculada para cada *frame* (sobreposto) do sinal original, permitindo assim a obtenção de um gráfico de leitura tri-dimensional onde o eixo vertical indica as faixas de frequências, o eixo horizontal representa o tempo e, por fim, as cores indicam a magnitude da amplitude (ou potência) do sinal em cada par tempo-frequência (18). A Figura 9 ilustra a representação de um sinal de áudio neste domínio tempo-frequência através de um espectrograma. As cores indicam a magnitude da amplitude do sinal em cada frequência e a cada instante de tempo.

### 2.4.2 Espectrogramas na Escala Mel

A percepção humana, tanto para a altura do som quanto para frequências distintas, segue um comportamento não linear. Considerando apenas o universo relacionado à frequência, seria extremamente interessante ter em mãos um atributo capaz de aproximar a análise dentro do universo da percepção humana em termos de escala. Uma das tratativas capaz de ser realizada para ete fim é a utilização de uma escala que possa simular a não linearidade humana de percepção: a escala Mel é uma das escalas que atendem essa necessidade (21).

Assim, considerando a escala Mel, é dito que a relação entre Hertz e Mel é aproximadamente linear para frequências abaixo de 500 Hz. Acima disso, a diferença de valores entre as duas escalas se torna cada vez maior. O ponto de referência da escala é 1000, indicando que 1000 Hz é igual a 1000 Mel (21). Em (22), a conversão entre Hertz e Mel é dada por:

$$Mel(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.18}$$

Analogamente, para converter elementos da escala Mel em Hertz, é possível aplicar:

$$Hertz(\text{Mel}) = 700\left(10^{m/2595} - 1\right) \tag{2.19}$$

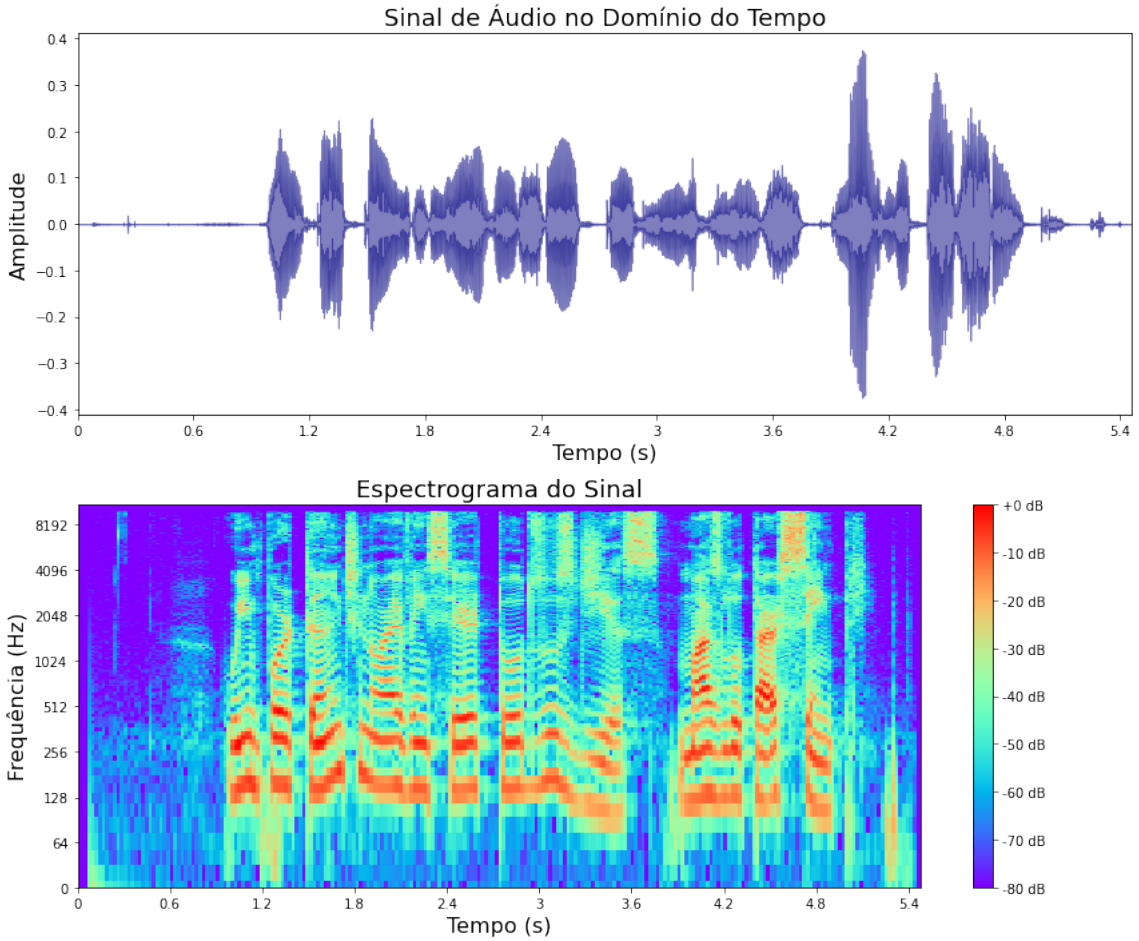


Figura 9 – Representação visual de um sinal de áudio no domínio do tempo (parte superior) e seu respectivo espectrograma (parte inferior).

Um passo essencial na construção do espectro de frequências na escala Mel é a construção de bancos de filtros triangulares responsáveis por essa transformação espectral. Em (23), é possível notar que, tipicamente, aplicações costumam utilizar 40 bandas (ou filtros) para essa transformação. Cada filtro no banco de filtros possui um formato triangular, retornando uma resposta de 1 na frequência central e diminuindo linearmente em direção a 0 até atingir as frequências centrais dos dois filtros adjacentes onde a resposta é 0, como mostrado na Figura 10.

Após a construção e aplicação do banco de filtros, é possível gerar um espectrograma com elementos já presentes na escala Mel. A matriz $S$ resultante da STFT, gerada para a representação de um espectrograma padrão, possui dimensões (amostras por frame/2, #frames). Já o resultado da aplicação do banco de filtros triangulares é uma matriz $M$ de dimensões (#bandas, amostras por frame/2). Dessa forma, o espectrograma na escala Mel é dado pela multiplicação matricial representada pela equação (2.20) abaixo e sua ilustração gráfica pode ser visualizada a partir da Figura 11 em uma visão comparativa

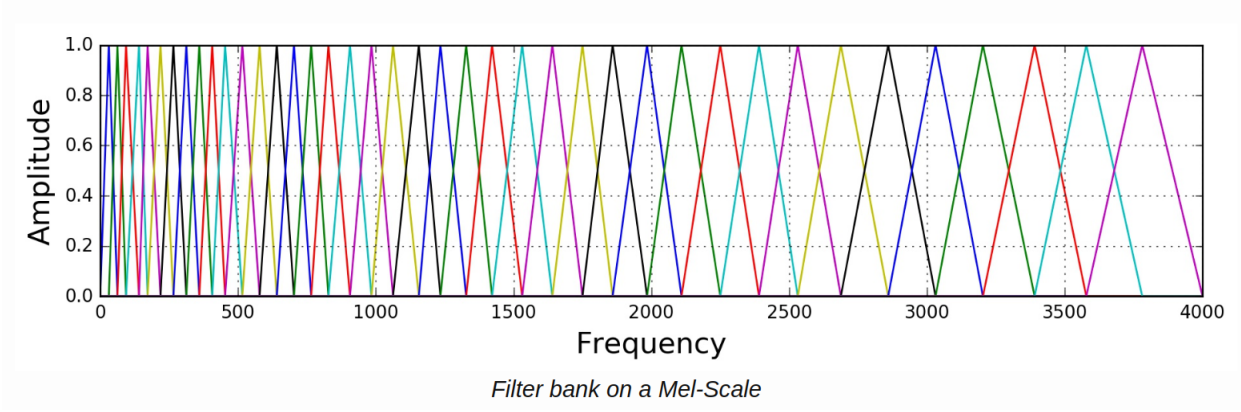


Figura 10 – Representação visual do banco de filtros triangulares utilizados na conversão de um espectrograma para a escala Mel.

entre um espectrograma detalhado na seção 2.4.1 e sua representação na escala Mel.

$$\text{Espectrograma Mel} = M \times S \tag{2.20}$$

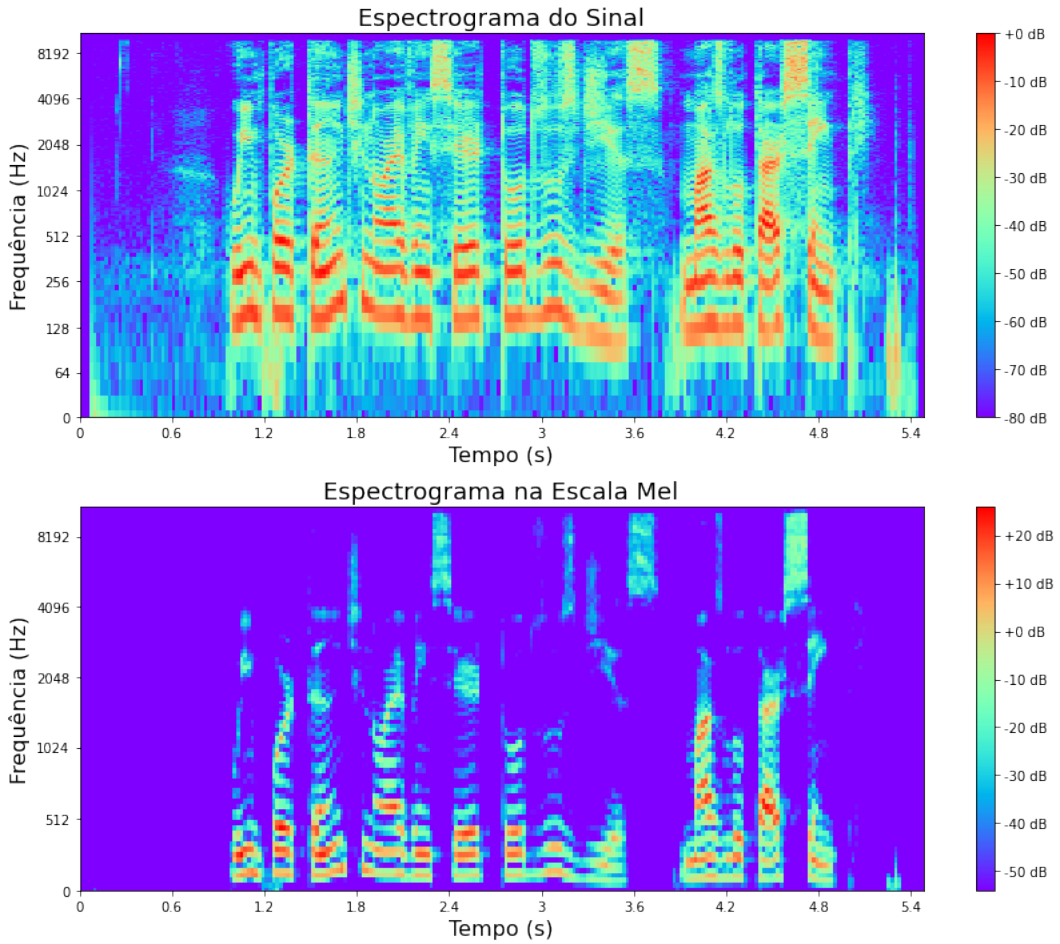


Figura 11 – Visualização de um espectrograma (acima) e sua representação na escala Mel (abaixo).

Na Figura 11, é possível notar significativas alterações no espectrograma proposto na escala Mel em relação ao espectrograma geral do sinal de áudio. Em linhas gerais, os componentes atrelados às baixas frequências ganham mais notoriedade na representação em escala Mel se comparados aos componentes das altas frequências. Além de motivações associadas à percepção humana de voz, essa concentração observada poderá ser discutida em maiores detalhes na seção seguinte, onde os MFCCs serão introduzidos.

### 2.4.3 MFCCs

Os MFCCs (do inglês *Mel-frequency cepstral coefficients*) são considerados elementos fundamentais em grande parte dos projetos relacionados a processamentos de sinais de áudio. Sua vasta aplicabilidade abrange um grande leque de aplicações, sendo algumas delas definidas em (4, 5, 7, 15, 22, 24).

Em (22), o fluxo de extração dos MFCCs é definido por:

- Aplicação da DFT no sinal de áudio amostrado
- Conversão do espectro de potências para a escala Mel
- Cálculo do logaritmo da potência para cada frequência na escala Mel
- Aplicação da Transformada Discreta de Cosseno (DCT, do inglês *Discrete Cosine Transform*)
- Definição final dos vetores das features (amplitudes do espectro resultante)

Os coeficientes MFCCs foram concebidos considerando a percepção e a sensibilidade humanas em diferentes faixas de frequências e, dessa forma, um dos passos para sua extração é a conversão do espectro de frequências gerado pela Transformada de Fourier para a escala Mel. Em (5), é dito que os MFCCs representam unidades distintas de som ou fonemas em seu formato natural, simulando e associando os procedimentos presentes no trato vocal humano dentro do contexto de geração e filtragem da voz. Por esse motivo, trata-se de uma excelente característica utilizada no entendimento de sinais de áudio e em aplicações que envolvem elementos da fala humana. A Figura 12 ilustra os resultados obtidos durante o cálculo dos coeficientes.

Durante as etapas do cálculo dos MFCCs, é aplicada a DCT no espectro de potências já convertido para a escala Mel. Considerando que a aplicação da DFT em um sinal no domínio do tempo resulta na representação deste sinal no domínio da frequência, ao aplicar a DCT em um espectro já consolidado no domínio da frequência, o resultado pode ser definido como uma representação *cepstral* em um ambiente conhecido como domínio da *quefrência*. Dessa forma, o *cepstrum* (intuitivamente dado como o resultado de um espectro) apresenta elementos próprios, como as *quefrências* e os *rhamonics*, os quais

podem ser filtrados utilizando procedimentos de *liftering*. Tais conceitos podem ser vistos em detalhes em (25).

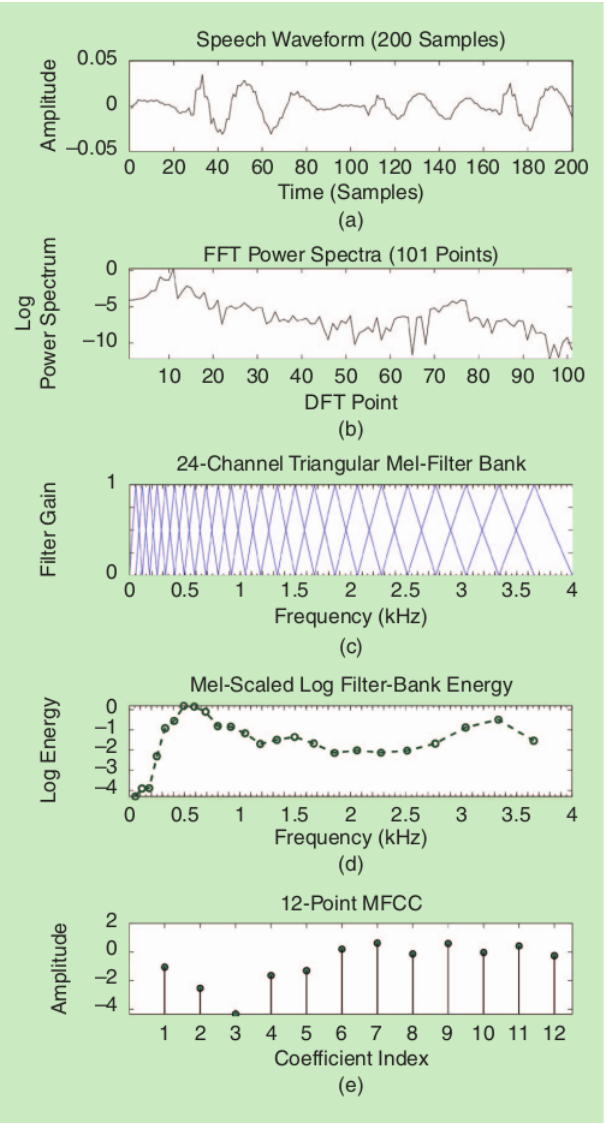


Figura 12 – Etapas de extração dos coficientes MFCCs de um sinal de fala: (a) Frame com 200 amostras representando 25 milissegundos do sinal original de fala amostrado a uma taxa de 8 kHz. (b) Espectro de potências gerado após a aplicação da DFT no sinal mostrando os primeiros 101 pontos. (c) Banco de filtros triangulares Mel de 24 canais. (d) Elementos de saída do espectro Mel após aplicação de transformação logaritma. (e) Primeiros 12 coeficientes estáticos MFCCs obtidos através da aplicação da DCT no espectro de energia.

Um fator importante sobre os MFCCs é que a transformação para o domínio *cepstral* é feita a partir de uma DCT e não de uma IDFT e, em (7), é possível observar que as DCTs possuem duas proprierdades relevantes neste aspecto: 1) é uma transformada que comprime a energia de um sinal em um número pequeno de coeficientes e 2) seus coeficientes resultantes são altamente descorrelacionados. Em outras palavras, pode-se dizer que a DCT auxilia no aumento da eficiência da modelagem e também atua positivamente

na geração de coeficientes com baixa correlação entre si, o que é principalmente relevante em termos de aprendizado de máquina.

Em (15), o cálculo dos MFCCs é dado por:

$$c_{n_c} = \sum_{m=1}^{M} [\log Y(m)] \cos\left[\frac{\pi n_c}{M}\left(m - \frac{1}{2}\right)\right] \tag{2.21}$$

onde $n_c$ representa o índice do coeficiente *cepstral*. Em (18) é dito que, em geral, os primeiros índices são os que carregam as principais informações de fonemas e de sinais falados. Em grande parte das aplicações, utiliza-se os coeficientes de índices entre 4 e 20. Em (21), por exemplo, é dito que, para aplicações relacionadas ao processamento de sinais musicais, são computados coeficientes de índices entre 13 e 25. Além disso, em (7) pontua-se que, em geral, parâmetros de velocidade e aceleração são comumente calculados ao longo dos *frames* e adicionados ao vetor final dos coeficientes MFCCs. Tais parâmetros, conhecidos como primeira e segunda derivadas, representam propriedades dinâmicas dos coeficientes.

Com isso, encerra-se o bloco de seções criado com o intuito de fornecer visões gerais sobre as principais características de sinais de áudio a serem utilizadas nas próximas seções em estudos mais aprofundados de modelagem. Após este feito, é possível iniciar descrições mais detalhadas sobre ferramentas capazes de proporcionar ao usuário formas dinâmicas e interativas de se extrair as características aqui citadas.

## 2.5 Biblioteca Python: librosa

Em uma visão geral, librosa é uma biblioteca Python construída para o processamento de áudio e sinais musicais. Seu conteúdo proporciona implementações de um conjunto variado de funções utilizadas nas mais diversas aplicações relacionadas ao processamento e análise de sinais (11).

As motivações por trás da criação de uma biblioteca desse porte tiveram seus alicerces baseados no crescente desenvolvimento observado no campo de pesquisa relacionado à MIR (*Music Information Retrieval*) e, considerando o Python como uma linguagem já consolidada em termos de processamento de dados, Aprendizado de Máquina e aplicações Web, sentiu-se a necessidade de construir um pacote capaz de entregar ferramentas básicas e avançadas para o processamento de áudio para as mais diferentes aplicações. O nome librosa é inspirado de LabROSA: *Laboratory for the Recognition and Organization of Speech and Audio* na Universidade de Columbia, onde o desenvolvimento inicial da biblioteca foi concebido (11).

A biblioteca librosa, em sua versão de referência dentro deste trabalho de graduação (0.8.0), possui uma série de métodos e funções capazes de entregar cálculos complexos

no âmbito de processamento de áudios. Muitos dos tópicos teóricos presentes nesta fundamentação podem ser calculados em poucas linhas de código, o que torna sua utilização extremamente atrativa em termos de construção de novas aplicações. No exemplo abaixo, extraído e adaptado de (11), a biblioteca librosa é utilizada para ler um sinal de áudio (originalmente dado no domínio do tempo), calcular a matriz espectral através da aplicação da STFT, calcular a matriz espectral na escala Mel com 40 bandas e retornar os 13 primeiros coeficientes MFCCs:

```
import librosa

filename = librosa.util_example_audio_file()
y, sr = librosa.load(filename, offset=25.0, duration=20.0)

spectrogram = librosa.stft(y=y, n_fft=2048, hop_length=512)

mel_spectrogram = librosa.feature.melspectrogram(y=y, n_mels=40)

mfccs = librosa.feature.mfcc(y=y, n_mfcc=13)
```

O módulo *display* da biblioteca librosa providencia interfaces para visualização de áudios renderizados e tratados através do *matplotlib* (biblioteca Python para plotagens gráficas). Um exemplo de uso desse método pode ser observado na própria página de documentação da librosa (26), com a Figura 13 demonstrando um espectrograma a partir da função *specshow*.

Em continuidade à definição de ferramentas e tópicos associados ao entendimento de sinais de áudio a partir da extração de suas características e posterior modelagem, é imprescindível introduzir conceitos relacionados ao Aprendizado de Máquina. Nesta seção, foi possível ter uma noção prática de como a linguagem Python e a biblioteca LibROSA fornecem um leque poderoso para o processamento de áudio. Nas próximas seções, será possível visualizar como essas características extraídas podem ser usadas para a criação de sistemas inteligentes.

## 2.6 O Aprendizado de Máquina

Ao longo dos últimos anos, o termo "aprendizado de máquina" (ou *Machine Learning*) tornou-se extremamente popular no âmbito da tecnologia. Muitas aplicações surgiram e diversas companhias se utilizaram das ferramentas do aprendizado de máquina para resolver problemas antes considerados complexos. Entretanto, esse universo ainda é considerado um tabu em alguns campos da tecnologia, muito disso devido a certa complexidade que se tem em entender, de fato, os conceitos que compõe a esfera do aprendizado de máquina.

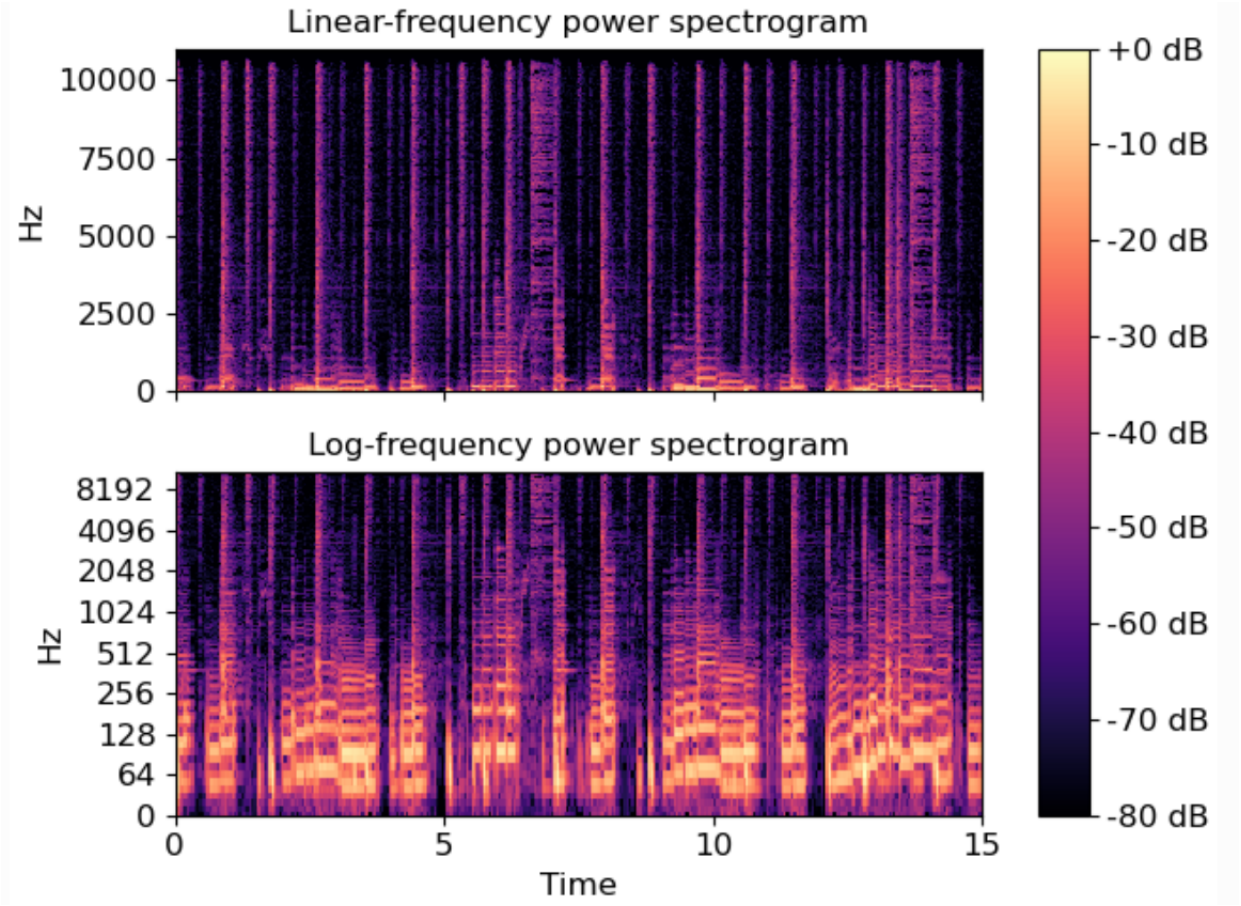


Figura 13 – Espectrograma em escala linear (parte superior) e logarítmica (parte inferior) utilizando a função specshow do módulo librosa.display.

De uma forma direta, *Machine Learning* é a ciência (e arte) relacionada à programação de computadores para que estes possam aprender utilizando dados (27). Neste mundo, algumas categorias foram criadas para caracterizar os problemas a serem resolvidos, gerando assim duas principais formas de se aplicar aprendizado de máquina:

- Aprendizado supervisionado
- Aprendizado não supervisionado

Andrew Ng defende em (28) que o **aprendizado supervisionado** é definido quando tem-se uma variável resposta predefinida na base de dados, permitindo assim com que o modelo a ser construído consiga se utilizar deste fator para associar e implementar regras matemáticas de modo a alcançar os resultados desejados. Já o **aprendizado não supervisionado** é concebido quando não existe uma variável resposta fornecida para determinada tarefa, exigindo assim uma modelagem baseada em associações próprias do modelo sem que este se utilize de um alvo definido previamente.

Existem ainda alguns outros contextos de aplicações de aprendizado de máquina, como por exemplo, modelos de recomendação, aprendizado por reforço, entre outros. Para

fins de direcionamento acordados com o objetivo desta fundamentação, os próximos tópicos definidos terão, como foco, as características de modelos de aprendizado supervisionado, mais especificamente modelos de classificação. De modo a propor uma visão geral sobre as principais aplicações onde o aprendizado de máquina se faz presente, a Tabela 1 detalha alguns elementos essenciais dentro de cada categoria deste universo.

Tabela 1 – Visão geral sobre aprendizado de máquina e sua aplicabilidade.

| Tipo de Aprendizado | Modelo | Exemplo de Aplicação |
|---|---|---|
| Supervisionado | Regressão | Previsão de preços de imóveis. |
| | Classificação | Classificação fraudes. |
| Não Supervisionado | Clusterização | Agrupamento de clientes por consumo. |
| | Redução de Dimensionalidade | Consolidação de variáveis para otimização de código. |

### 2.6.1 Aprendizado Supervisionado: Classificação

Como mencionado anteriormente, em um contexto de aprendizado supervisionado, o treinamento de modelos é baseado em uma resposta conhecida. Dessa forma, é necessário fornecer ao algoritmo um conjunto $x^{(i)}$ de entradas e uma coleção $y^{(i)}$ com as saídas associadas. Nesse meio, é possível dividir as aplicações de aprendizado supervisionado em duas principais tarefas: regressão e classificação.

Modelos de **regressão** estão associados a saídas $y$ contínuas, ou seja, o resultado de um modelo treinado nesse contexto diz respeito a um valor numérico absoluto que pode ser utilizado para decisões quantitativas relacionadas a volume, *forecast*, entre outros. Já em modelos de **classificação**, a saída $y$ tem um caráter discreto, referindo-se assim a valores qualitativos utilizados para decisões categóricas (sim ou não, aprovado ou reprovado, fraude ou não fraude). No limite, é possível configurar alguns modelos de classificação para que estes retornem um valor numérico representando a probabilidade de uma determinada amostra pertencer a uma determinada classe. A Figura 14 traz uma visão geral sobre a principal diferença entre modelos de regressão e classificação em termos de critérios de decisão. Neste trabalho, será dada ênfase em modelos de classificação, sendo alguns dos principais algoritmos neste meio representados por:

- Regressão Logística
- Árvores de Decisão
- Florestas Aleatórias
- Máquina de Vetores Suporte
- k-Vizinhos Mais Próximos

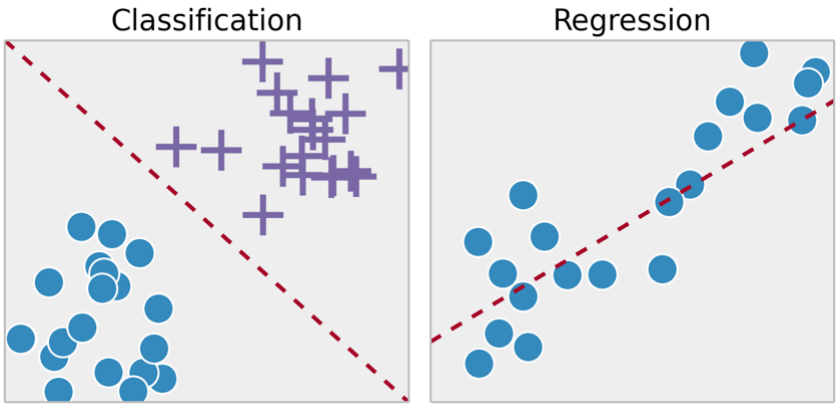


Figura 14 – Exemplificação de modelos de aprendizado supervisionado: classificação (à esquerda) e regressão (à direita).

De um modo geral, o treinamento de modelos de classificação envolve uma série de definições e fatores dependentes do algoritmo a ser utilizado. Em uma Regressão Logística, por exemplo, é definida uma função sigmoidal para realizar a separação do conjunto de dados em um universo multidimensional, sendo sua otimização realizada por uma função custo que permite encontrar os melhores parâmetros para o menor erro possível durante a etapa de treinamento. Por outro lado, uma Árvore Decisão leva em consideração a melhor separação do conjunto de dados em termos como "nós", "folhas", "entropia", entre outros. O algoritmo SVM (abreviação para *Support Vector Machines*) tem como base a definição do melhor hiperplano capaz de separar as classes de um conjunto de dados.

Em (27) é possível encontrar exemplos práticos de modelos de classificação. Em (29, 30) é possível encontrar informações relevantes sobre a biblioteca Python *scikit-learn* utilizada em larga escala para esse tipo de aplicação. Por fim, em (31), tem-se uma vasta documentação da biblioteca *scikit-learn* direcionada para modelos de classificação, permitindo assim uma análise mais detalhada sobre as nuances e características dos pricipais algoritmos desenvolvidos para esse fim.

### 2.6.2 Métricas em Modelos de Classificação

Como pôde ser visto, existem diferentes formas de se treinar um algoritmo de *Machine Learning*, mesmo restringindo estes ao domínio de modelos de classificação (aprendizado supervisionado). Na prática, de forma genérica, o resultado do treinamento de modelos de classificação envolve a possibilidade de gerar saídas discretas ou prever classes de acordo com um conjunto de variáveis de entradas. Após essa etapa, é extremamente importante encontrar formas de validar se esse treinamento foi realizado de forma correta e se o modelo gerado pode ser utilizado para resolver a tarefa ao qual foi especificado. Para as definições a seguir, foram consideradas as referências (27, 32, 33, 34, 35).

#### 2.6.2.1 Matriz de Confusão

Para verificar o desempenho de um modelo de classificação já treinado, é possível utilizar métricas bem definidas aplicadas de acordo com o contexto da modelagem. Iniciando essa jornada de definição, será abordado neste tópico o conceito atrelado à matriz de confusão, com a ideia geral vinculada à contagem de elementos pertencentes a uma classe A que foram classificados pelo modelo como parte de uma classe B (27).

Simplicando o contexto de modelagem a um modelo de classificação binária, o entendimento da matriz de confusão parte da definição dos termos que a compõem. Para este fim, a Tabela 2 traz uma visão geral sobre estes termos e seus respectivos significados, onde $y_{true}$ refere-se à variável resposta oficial presente na base de treinamento e $y_{pred}$ indica a classe predita pelo modelo treinado.

Tabela 2 – Definições dos termos de uma matriz de confusão.

| Descrição | Abrev | Definição |
|---|---|---|
| True Positives | TP | $y_{true} = 1$; $y_{pred} = 1$ |
| False Positives | FP | $y_{true} = 0$; $y_{pred} = 1$ |
| True Negatives | TN | $y_{true} = 0$; $y_{pred} = 0$ |
| False Negatives | FN | $y_{true} = 1$; $y_{pred} = 0$ |

Assim, uma matriz de confusão pode ser considerada a representação visual da disposição dos elementos definidos na Tabela 2 de modo a proporcionar uma ideia de desempenho sobre os resultados de um classificador treinado. A Figura 15 ilustra uma matriz de confusão gerada para um modelo de classificação binário. De um modo geral, espera-se que um bom modelo tenha um alto volume vinculado aos elementos $TP$ e $TN$ dado que estes indicam, respectivamente, a quantidade de elementos da classe positiva classificados corretamente como sendo da classe positiva e a quantidade de elementos da classe negativa corretamente classificados como sendo da classe negativa.

Considerando a definição da matriz de confusão e dos elementos que a compõem, é possível pontuar que as próximas métricas a serem definidas têm estes elementos como base matematicamente dispostos em diferentes formatos, gerando assim métricas específicas que podem ser utilizadas em diferentes aplicações.

#### 2.6.2.2 Acurácia

A acurácia de um modelo pode ser entendida como a quantidade de predições corretas em relação ao total de elementos classificados. Matematicamente, é possível definir a acurácia de um modelo de classificação como (27):

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.22}$$

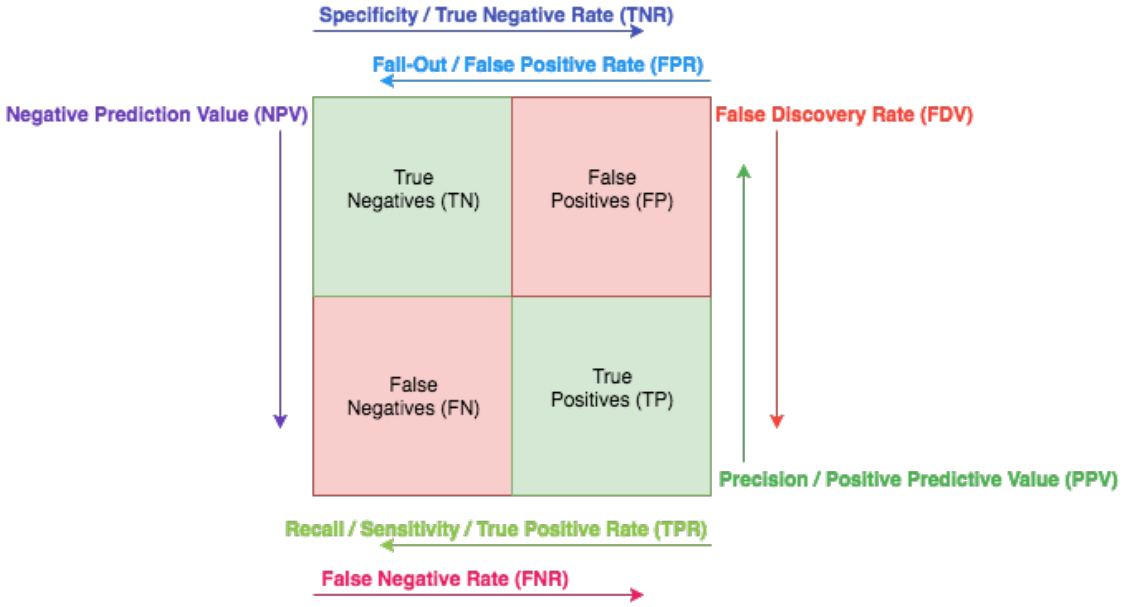


Figura 15 – Exemplo de disposição dos elementos de avaliação em uma matriz de confusão.

No numerador da equação (2.22), tem-se a quantidade de elementos classificados corretamente pelo modelo em ambas as classes (positiva ou negativa). Já no denominador, tem-se a soma de todos os elementos envolvidos, indicando assim que a acurácia é uma forma de verificar o percentual de acerto de um modelo treinado.

Entretanto, apesar de promover um fácil e intuitivo entendimento, nem sempre a acurácia é considerada uma boa métrica de avaliação para modelos de classificação. Em (27, 28), é possível encontrar situações onde a acurácia pode fornecer uma falsa ideia de bom desempenho, sendo o principal cenário onde isso acontece descrito por bases com a classe *target* altamente desbalanceada. Em um exemplo prático, pode-se considerar a tarefa de construção de um modelo de predição de transações fraudulentas onde, no conjunto de dados fornecido, tem-se apenas 1% de fraudes contra 99% de transações consideradas corretas. Neste caso, uma simples regra que nunca prevê fraude, isto é, retorna que a transação sempre será correta, acertará em 99% dos casos, apresentando uma acurácia de mesmo valor. No mundo real, este é um cenário altamente indesejado e, de modo a propor uma forma de contornar isso, outras métricas foram desenvolvidas.

#### 2.6.2.3 Precisão

A precisão de um modelo de classificação é um indicativo do quão preciso e acertivo é o modelo treinado. Matematicamente, é definida por (27):

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{2.23}$$

Analisando a equação (2.23), percebe-se que a precisão é calculada através de uma

divisão entre a quantidade de elementos da classe positiva classificados corretamente (TP) e a quantidade total de elementos classificados como sendo da classe positiva (TP + FP). De modo geral, a precisão responde a seguinte pergunta: "De todas as predições da classe positiva, quantas realmente são da classe positiva?".

Intuitivamente, aumentar a precisão de um modelo significa reduzir o número de Falsos Positivos (ex: transações normais classificadas como fraude ou vozes de impostores classificadas como pessoas legítimas). Contudo, nem sempre este é o melhor cenário exigido pelo contexto de modelagem e, por este motivo, a precisão está sempre vinculada a uma outra métrica de avaliação: o recall ou revocação.

#### 2.6.2.4 Recall

O recall de um modelo traz uma ideia de taxa de verdadeiros positivos (ou *True Positives Rate*) e responde a seguinte pergunta: "De todas as instâncias pertencentes a classe positiva, quantas foram classificadas corretamente?". Matematicamente, o recall é dado por (27):

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.24}$$

Comparando com a equação 2.23, pode-se dizer que a equação 2.24 dá maior importância aos Falsos Negativos (ex: transações fraudulentas classificadas como transações normais). Na prática, o recall traz uma relação entre os elementos da classe positiva classificados corretamente (TP) sobre os elementos que deveriam ser classificados como sendo da classe positiva (TP + FN). Em um cenário próximo ao contexto deste trabalho de graduação, utilizar o recall como métrica de otimização significaria reduzir a quantidade de vozes legítimas classificadas como vozes impostoras.

Neste cenário, percebe-se que existe uma relação entre a precisão e o recall de modelos treinados. Maximizar um significa automaticamente diminuir o outro. A esta dependência dá-se o nome de *Precision-Recall trade off*.

#### 2.6.2.5 F1-Score

De modo a mitigar essa relação de troca entre precisão e recall, a métrica F1-score é concebida como uma média harmônica destes dois termos (27). Matematicamente, o Score F1 é dado por:

$$\text{F1-score} = 2 \times \frac{\text{Precisão} * \text{Recall}}{\text{Precisão} + \text{Recall}} \tag{2.25}$$

A métrica F1-score pode ser utilizada como objeto de minimização, visto que penaliza modelos onde a diferença entre precisão e recall é muito ampla, retornando bons

índices caso os dois sejam próximos e em níveis razoáveis.

#### 2.6.2.6 Log Loss

A métrica *Log Loss* (ou perda logarítmica) pode ser definida, em particular, quando a saída de um modelo de classificação é dada por uma probabilidade e não pela definição categórica e discreta de uma classe. Matematicamente, é possível obter o desempenho de um classificador por perda logarítmica utilizando a relação abaixo:

$$\text{log-loss} = \sum_{i=1}^{T} y_i \log(p_i) + (1 + y_i) \log(1 - p_i) \tag{2.26}$$

onde $T$ representa a quantidade total de registros da base de dados avaliada, $y_i$ refere-se ao rótulo para a instância de índice $i$ e $p_i$ representa a probabilidade retornada pelo classificador para a classe positiva.

#### 2.6.2.7 Curva ROC e AUC

A curva ROC (abreviação para *Receiver Operating Characteristic*) é extremamente útil para visualizar o balanceamento entre a Taxa de Verdadeiros Positivos (TPR ou Recall) e a Taxa de Falsos Positivos (FPR ou *Sensitivity*) ao longo de diferentes limiares de classificação. Para visualizar esse conceito, a Figura 16 mostra a curva ROC para diversos classificadores já treinados.

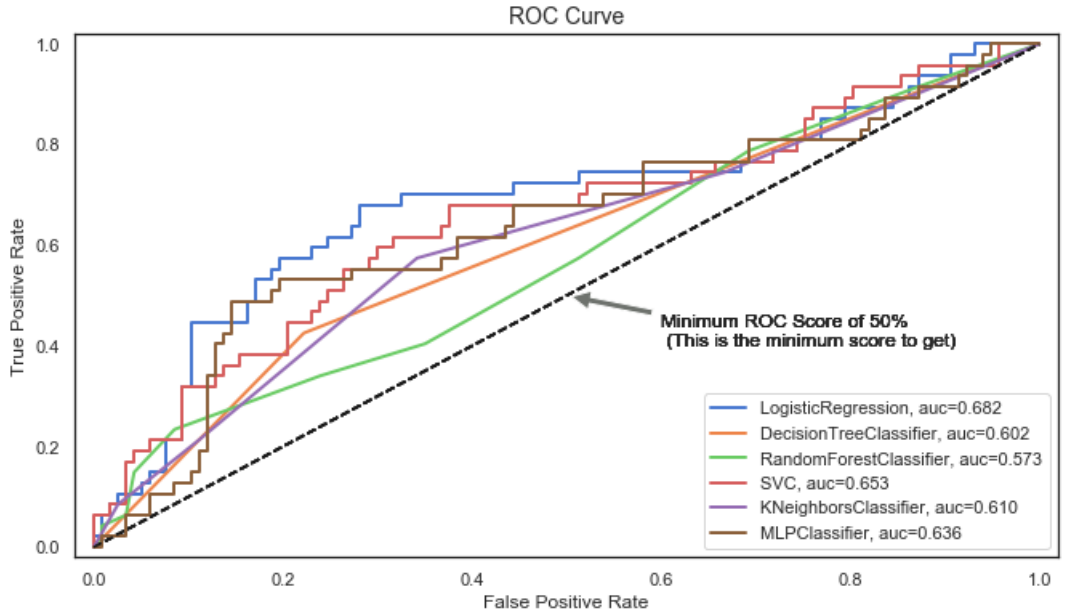


Figura 16 – Curva ROC para classificadores treinados.

A comparação de desempenho entre classificadores utilizando a curva ROC se dá através de uma outra métrica conhecida como AUC (abreviação para *Area Under the Curve*)

onde, de fato, é calculada à área abaixo da curva ROC, tendo seu valor analisado como uma métrica de desempenho dos modelos. Em linhas gerais, supondo uma ordenação das amostras do modelo por probabilidade de pertencer à classe positiva, isto é, amostras com menores probabilidades posicionadas à esquerda e amostras com maiores probabilidedes à direita, um alto valor para a métrica AUC indica que são maiores as chances de encontrar uma amostra pertencente à classe positiva próxima às amostras ordenadas à direita (altas probabilidades).

## 2.7 Identificação de Locutores através da Fala

O reconhecimento de locutores através da fala (muitas vezes chamado de *speaker recognition*) é uma frente utilizada em diversas áreas como, por exemplo, saúde, robótica, controle de veículos, entre outros. Nos últimos anos, diversos sistemas de reconhecimento utilizando a fala foram desenvolvidos para sanarem os mais variados problemas do mundo real (22). Por sua origem biológica, a voz é considerada um importante elemento em aplicações biométricas e seu estudo compõe formas de construir aplicações relevantes nesse âmbito. A Tabela 3 proporciona uma análise sobre as principais tecnologias de biometria e seu desempenho em termos de acurácia, facilidade de uso, aceitação, facilidade de implementação e custo (24).

Tabela 3 – Comparação de diferentes características biométricas (* indica a facilidade de).

| Biometria | Acurácia | Utilização* | Aceitação | Implementação* | Custo |
|---|---|---|---|---|---|
| Voz | Média | Alta | Alta | Alta | Baixo |
| Face | Baixa | Baixa | Alta | Média | Baixo |
| Iris | Média | Média | Média | Média | Alto |
| Impressão Digital | Alta | Média | Baixa | Alta | Médio |
| Retina | Alta | Baixa | Baixa | Baixa | Médio |
| Geometria da Mão | Média | Alta | Média | Média | Alta |
| Assinatura | Média | Média | Alta | Baixa | Médio |

Ainda em (24), é dito que, conforme as informações dispostas na Tabela 3, a voz é uma das mais promissoras tecnologias, dada sua facilidade de utilização, implementação, aceitação por parte dos usuários e, por fim, seu baixo custo de geração. Em complemento, em (15) é dito que dois indivíduos não possuem vozes idênticas por conta das dimensões do trato vocal, tamanho da laringe e outros órgãos relacionados à produção da voz propriamente dita. Além dessas características físicas, cada locutor possui seu próprio modo de falar, incluindo sotaques, ritmo, entonação, pronúncia, padrão, vocabulário, entre outros.

Dessa forma, existe uma série de considerações a serem realizadas em relação à utilização da voz na construção de sistemas inteligentes. Em (7), pontua-se que, diferente

de outras formas de autenticação biométrica, a fala humana pode ser considerada uma biometria de desempenho. Em outras palavras, a informação de identidade do locutor está embutida, primariamente, em como a fala é pronunciada e não necessariamente no que é falado. Isso faz com que sinais de voz sejam propensos a um alto grau de variabilidade: mesmo uma única pessoa não pronuncia as mesmas palavras exatamente da mesma forma.

### 2.7.1 Definições e Conceitos Técnicos

Em (7) é possível encontrar uma vasta referência relacionada ao reconhecimento de locutores através da fala, desde motivações para uso dessa tecnologia até uma evolução dos conceitos técnicos aplicados ao longo do tempo. Entre os conceitos detalhados, é possível citar:

- **GMM:** Gaussian Mixture Models
- **GMM-UBM:** Gaussian Mixture Models com modelo universal (UBM, do inglês *Universal Background Model*
- **GMM-SVM:** Gaussian Mixture Models com Support Vector Machines
- **JFA:** Joint Factor Analysis
- **i-Vector:** União entre os métodos JFA e GMM-SVM
- **DNN:** Deep Neural Networks

Em (15), uma visão geral sobre o reconhecimento de voz em aplicações independentes de texto é fornecida, além da definição formal de conceitos essenciais que incluem o VAD (Voice Activity Detector), normalização de *features*, *supervectors*, entre outros tópicos também presentes em (7).

Os conceitos abordados em (14) serviram como base para muitos outros artigos publicados posteriormente, sendo esta uma importante referência para o entendimento dos primórdios da utilização da fala para o reconhecimento de locutores.

O conteúdo presente em (22) permite obter uma referência clara e direta sobre uma aplicação estritamente próxima dos objetivos deste trabalho de graduação, visto que sua implementação tem como base a utilização dos coeficientes MFCCs de um sinal de áudio para a identificação de locutores.

Assim, pode-se dizer que os artigos (7, 15, 24, 22, 14) podem ser considerados como referências técnicas e teóricas dentro dos objetivos deste trabalho de graduação, visto que trazem consigo fundamentações explicativas e detalhadas sobre os principais conceitos aplicados no universo de utilização de fala para a construção de sistemas inteligentes. O entendimento e a absorção do conhecimento presente nos artigos acima citados é extremamente importante para o desenvolvimento de aplicações relacionadas.

### 2.7.2 Métricas de Avaliação de Modelos

Assim como existem diferentes formas de se construir modelos de identificação de locutores a partir da fala, há diversas formas de avaliar se o sistema criado possui uma boa performance. Na fundamentação teórica deste trabalho de graduação, foram abordados conceitos de avaliação de modelos de aprendizado de máquina considerando implementações clássicas de modelos de classificação. Entretanto, para o reconhecimento de fala, existem alguns trabalhos publicados que utilizam métricas relativamente diferentes das métricas já abordadas.

Em (7) é possível obter um entendimento claro sobre os tipos de erros associados à identificação de locutores, bem como sobre as principais métricas utilizadas. Como definição inicial, é possível separar os erros em:

- **False-Acceptance (FA):** quando é concedido acesso a um locutor impostor
- **False-Rejection (FR):** quando o acesso a um locutor legítimo é bloqueado

Assim, define-se então as taxas de falsa aceitação (FAR, do inglês *False Acceptance Ratio*) e de falsa rejeição (FRR, do inglês *False Rejection Ratio*) por:

$$\text{FAR} = \frac{\text{Número de erros FA}}{\text{Número de tentativas de impostores}} \tag{2.27}$$

$$\text{FRR} = \frac{\text{Número de erros FR}}{\text{Número de tentativas legítimas}} \tag{2.28}$$

Dentro deste contexto, é possível definir a *taxa de erro igual* (ou EER - *Equal Error Rate*) quando os valores de FAR e FRR se tornam iguais a partir da modificação do limiar de aceitação (*threshold*) do modelo. A métrica EER é bem popular em sistemas de verificação de locutores; porém, em alguns casos, o limiar de aceitação definido para a EER pode não ser ideal para algumas aplicações. A Figura 17 ilustra as métricas definidas há pouco (7).

Adicionalmente, o Instituto Nacional e Padronização e Tecnologia dos Estados Unidos (NIST, do inglês *National Institute of Standards and Techniques*) providencia uma série de parâmetros e métricas padronizadas para as mais vastas aplicações. Em termos de detecção de locutores através da fala, além da visualização do *equal error rate* (EER), é dito que a inspeção visual da curva *detection error trade-off* (DET) também pode ser considerada uma abordagem comum em diversas aplicações (15). A Figura 18 traz uma ilustração comparativa entre dois sistemas através da curva DET. Considerando os tópicos sobre o limiar de decisão para a EER expostos acima, houve a necessidade de construir uma nova função responsável por avaliar dinamicamente o custo dos modelos de detecção

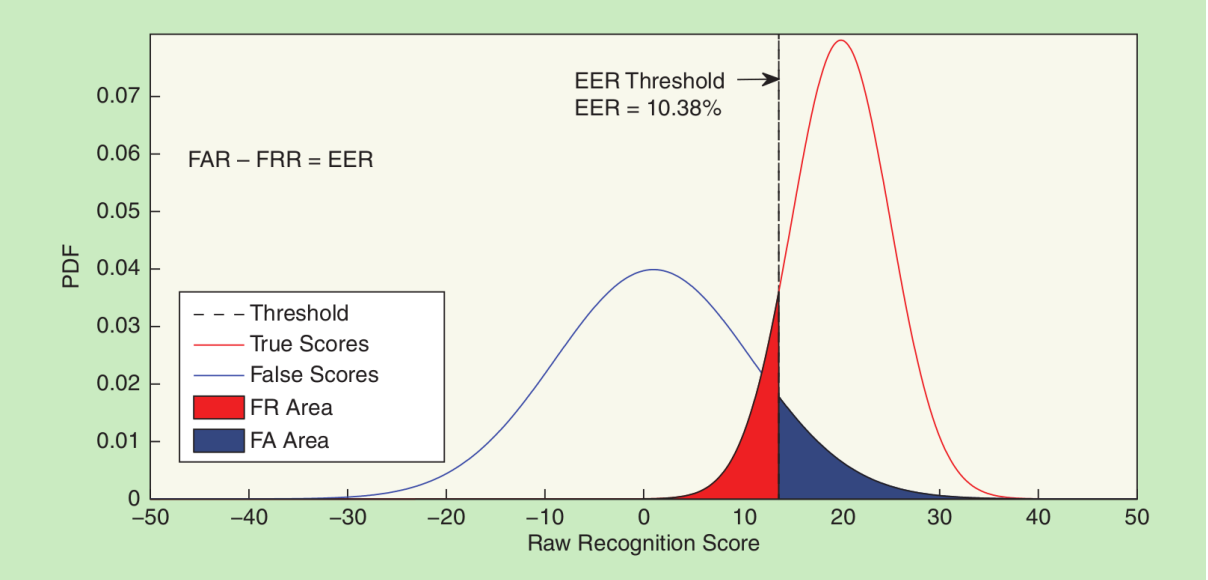


Figura 17 – Ilustração de curvas de distribuição de score para locutores *target* e não-target em decisões baseadas em um limiar de aceitação. As áreas abaixo das curvas azul e vermelha indicam, respectivamente, os erros FAR e FRR.

de locutores e, dessa forma, o NIST indica a utilização de uma *função custo de detecção* (DCF, do inglês Detection Cost Function) definida por:

$$\mathrm{DCF}(\tau) = C_{miss} P_{miss}(\tau) P_{target} + C_{FA} P_{FA}(\tau)(1 - P_{target}) \tag{2.29}$$

Onde:

- $C_{miss}$ = Custo de um erro FR
- $C_{FA}$ = Custo de um erro FA
- $P_{target}$ = Probabilidade de acerto para o locutor target
- $P_{miss}(\tau)$ = Probabilidade de erro dado o target para o limiar/*threshold* $\tau$
- $P_{FA}(\tau)$ = Probabilidade de erro dado o não-target para o limiar/*threshold* $\tau$

Em 2008, o NIST padronizou constantes para alguns parâmetros, configurando assim os termos $C_{miss} = 10$, $C_{FA} = 1$, $P_{target} = 0.01$. Em termos práticos, isso significa penalizar o sistema dez vezes mais em caso de erro de rejeição em relação ao erro de aceitação. Assim, a equação 2.29 pode ser reduzida a:

$$\mathrm{DCF}(\tau) = 0.1 \times P_{miss}(\tau) + 0.99 \times P_{FA}(\tau) \tag{2.30}$$

A partir do cálculo da DCF, é possível derivar duas métricas de desempenho muito utilizadas em problemas relacionados: a mínima DCF (MinDCF) e a DFC real (ActDCF). O termo MinDCF está associado ao valor mínimo da DFC a partir de alterações no

*threshold*. Ao otimizar o limiar e aplicar o modelo a um conjunto de validação, obtém-se então a métrica ActDCF (7, 15).

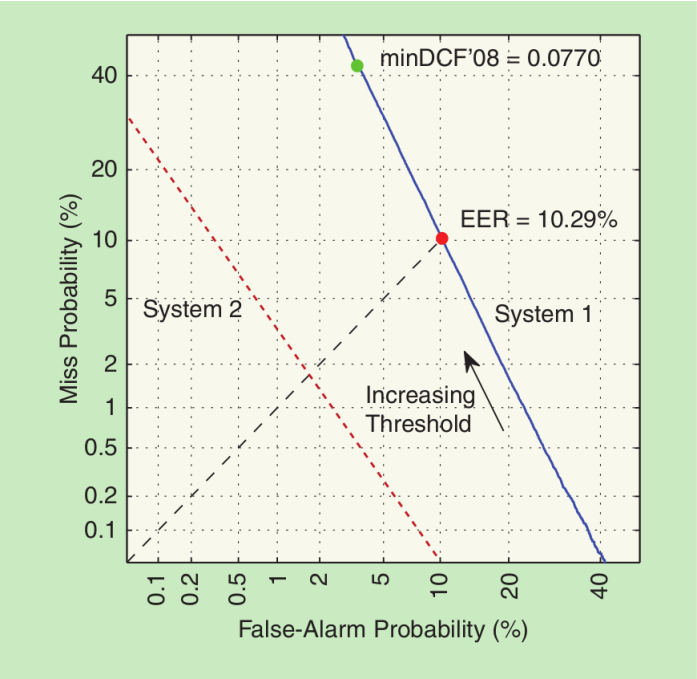


Figura 18 – Ilustração de curvas de detecção de erros (DET) para dois sistemas diferentes de identificação de locutores a partir da fala. No primeiro sistema (indicado por System 1), são mostrados os pontos correspondentes na curva que produzem o EER e o DCF mínimos (conforme o instituto NIST), e a direção crescente de variação do *threshold*. Por estar mais próximo da origem, o segundo sistema (indicado por System 2) mostra um melhor desempenho.

### 2.7.3 Trabalhos Relacionados

Assim, tendo em mãos um vasto conteúdo técnico e teórico sobre a aplicabilidade de sistemas inteligentes de reconhecimento de locutores a partir da voz, a análise de projetos já aplicados se faz extremamente necessária para que se possa ter insumos suficientes para o desenvolvimento a nível comparativo.

Em (13), um modelo de reconhecimento foi construído utilizando GMMs e uma base UBM para modelagem universal. Além disso, utilizou-se técnicas de normalização de score e a medição de desempenho através da razão de verossimilhança (*likelihood ratio*). Curvas DET foram obtidas para comparar os experimentos presentes com padrões estabelecidos pelo NIST.

De forma similar, é possível observar em (8) a utilização do conceito de combinação dos *supervectors* GMMs com o algoritmo SVM para a construção de um sistema capaz de reconhecer locutores através da voz. Como base de dados, foi utilizado o *corpus* disponibilizado pelo NIST contendo elementos de fala através do telefone. Entre as características de áudio extraídas para esta tarefa, foram considerados os primeiros 19

componentes dos MFCCs utilizando a função Hann como forma de suavização dos *endpoints* em *frames* de 10 ms. Além disso, a primeira derivada dos MFCCs foi computada para enriquecimento do vetor de variáveis. Adicionalmente, o algoritmo SVM foi configurado de modo a propor a remoção de sub-espaços que causam variabilidade no kernel (SVM NAP - *Nuisance Attribute Projection*). Nesse cenário, foi possível obter um melhor desempenho em relação a outros contextos comparativos utilizados no estudo como, por exemplo, modelos GMM UBM puros.

Já em (22), um modelo de detecção de locutores foi desenvolvido utilizando, como base, a extração dos MFCCs de sinais de áudio utilizando o software MATLAB, além de uma comparação com modelos baseados em redes neurais profundas. Como resultado, foi possível observar um bom desempenho da modelagem utilizando MFCCs em relação a técnicas gerais comparadas no estudo.

Em um formato compacto, é possível encontrar em (24) uma excelente relação de projetos realizados na última década envolvendo o reconhecimento de voz. Em (15), uma tabela comparativa entre alguns dos métodos de aplicação também é disponibilizada e pode ser utilizada como base para este trabalho de graduação.

# 3 Metodologia

O presente trabalho de graduação pretende aplicar o processamento e a análise de sinais de áudio para o treinamento de um modelo de aprendizado de máquina capaz de reconhecer locutores a partir de suas respectivas vozes. A caracterização do processo de modelagem envolve, em uma primeira etapa, a construção de uma base de dados através da gravação de áudios de três diferentes locutores seguindo um processo reproduzível e uniforme. Considerando a aplicabilidade universal do sistema desenvolvido e sua utilização por usuários externos, serão adicionados áudios oriundos de uma base pública contendo sinais gravados e validados de locutores genéricos (UBM).

Em posse da base de dados devidamente etiquetada para cada um dos locutores autênticos e, adicionados os elementos da base pública de locutores genéricos, pretende-se então aplicar um rico processo de extração de características das vozes gravadas para a retirada de significado computacionalmente interpretável. Nesta etapa, serão consideradas agregações estatísticas de características de sinais de áudio, muitas destas presentes em (21). Com isso, será possível gerar uma matriz multidimensional a ser utilizada como insumo para o treinamento de algoritmos de aprendizado de máquina que, por sua vez, têm como principal objetivo o mapeamento de qualquer novo sinal de entrada a um dos locutores presentes na base, incluindo a classificação genérica. A Figura 19 traz uma visão geral sobre o diagrama de desenvolvimento do trabalho, permitindo assim uma análise detalhada sobre todas as etapas presentes na construção do projeto.

Como métrica de otimização da etapa de modelagem, definiu-se a precisão como uma forma de garantir um aprimoramento específico visando a diminuição de vozes impostoras (classe de locutores genéricos) classificadas como legítimas (locutores cujas vozes foram coletadas pontualmente). Em uma possível aplicação real, o custo de liberar um acesso a um impostor em um processo de autenticação por voz seria relativamente maior do que recusar um acesso de alguém legítimo. Assim, a métrica de precisão atende perfeitamente este objetivo.

Assim, na seção seguinte, serão pontuadas as etapas relacionadas à criação da base de dados utilizada nos processos de extração de características de áudio e posterior modelagem.

## 3.1 Construção da Base de Dados

Em praticamente todas as aplicações que envolvem o treinamento de algoritmos de aprendizado de máquina, é dito que os dados são o componente principal para que se possa

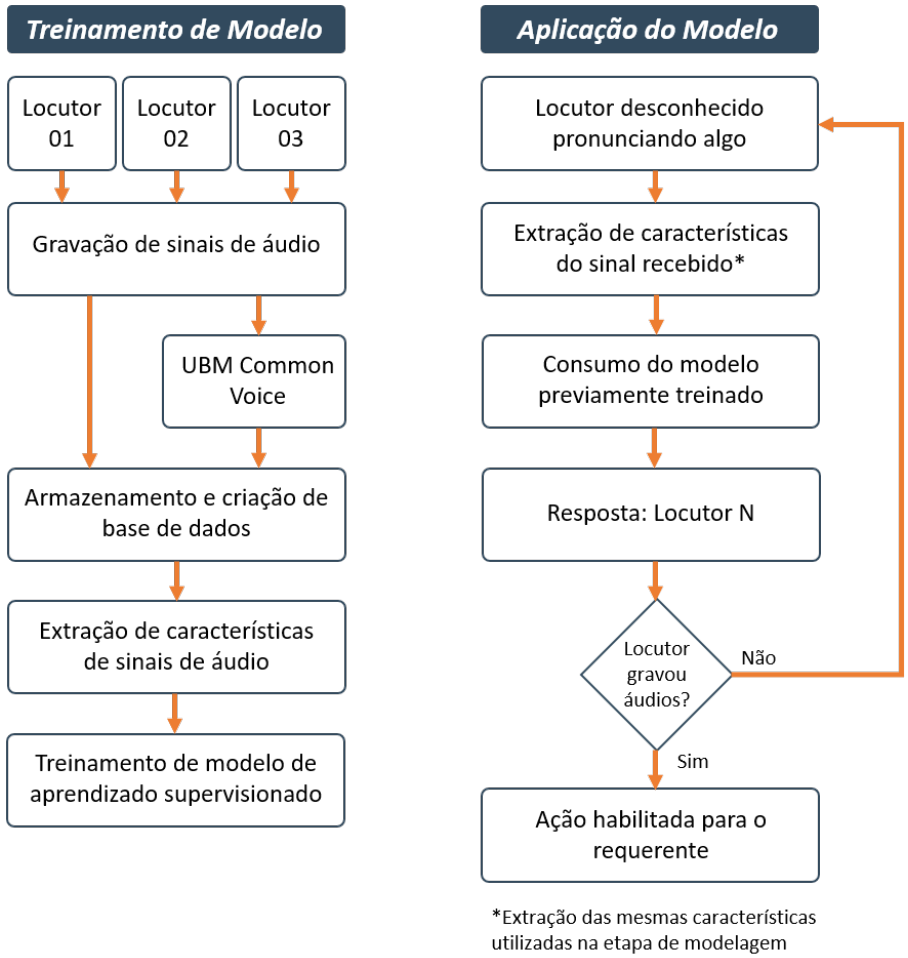


Figura 19 – Diagrama de desenvolvimento e aplicação do projeto. À esquerda, são detalhadas as etapas necessárias para a construção da inteligência de reconhecimento de locutores a partir das vozes. À direita, tem-se um exemplo de fluxo de aplicação da inteligência desenvolvida para o permissionamento de ações de acordo com o resultado do modelo.

construir modelos eficientes. Em linha com as diretrizes deste trabalho, os próximos tópicos trarão detalhes relevantes sobre todo o processo de construção da base de dados utilizada no projeto, desde o método de gravação até a estruturação dos dados relacionados.

### 3.1.1 Gravação de Áudios pelos Locutores

Considerando a proposta e os objetivos do referido trabalho, o processo de construção da base de dados teve como princípio a uniformidade e reprodutibilidade das gravações das vozes de cada um dos locutores protagonistas do modelo de reconhecimento. Nesse cenário, foram estabelecidas as seguintes diretrizes para a obtenção de áudios válidos a serem utilizados na etapa de treinamento:

- Utilização de um mesmo hardware para todos os locutores
- Gravação em ambiente silencioso

- Pronúncia clara e contínua de frases curtas a cada gravação
- Armazenamento dos áudios em um mesmo formato para todos os locutores

Em uma necessidade de simplificação do processo de obtenção dos dados e, considerando uma maior facilidade na aplicação prática do modelo resultante, decidiu-se utilizar, para a gravação dos sinais de áudio, um *smartphone* de geração intermediária com especificações mostradas na Tabela 4.

Tabela 4 – Definições do hardware utilizado na gravação dos áudios.

| Smartphone | Modelo | Sistema Operacional | Memória RAM | Memória ROM |
|---|---|---|---|---|
| Motorola | Moto g(6) plus | Android 9 | 4,00 GB | 64,00 GB |

Visando propor um meio dinâmico de gravação de sinais de áudio, utilizou-se o aplicativo gratuito *Voice Recorder* (36), disponível para Android e obtido através da *Google Play Store*. A escolha por este aplicativo teve como base a simplicidade da interface e as funcionalidades disponíveis, permitindo assim a fácil gravação e o armazenamento de sinais de áudio, além de uma rota de descarte de arquivos gravados, caso estes não cumprissem os requisitos estabelecidos (ruído externo ou fluidez de pronúncia do locutor). Entre as principais características do aplicativo, considerando o período de referência deste trabalho, é possível citar:

- Mais de 10 milhões de downloads e 880 mil avaliações
- Funcionalidades *play, pause* e *stop* de áudio
- Possibilidade de envio e compartilhamento de arquivos
- Possibilidade de eliminação de arquivos direto no aplicativo
- Codificação mp3 com taxa de amostragem ajustável (8–44 kHz)
- Ferramenta de microfone com calibração de ganho

A Figura 20 mostra três exemplos de telas extraídas do aplicativo *Voice Recorder* onde é possível identificar, por exemplo, a página utilizada para a gravação dos sinais de áudio, a sessão relacionada ao armazenamento dos arquivos gravados e também a tela de configurações do aplicativo com parâmetros de gravação estabelecidos.

De forma individual, cada locutor, em posse do dispositivo móvel especificado na Tabela 4, realizou sucessivas gravações através da pronúncia de frases curtas, distintas e previamente estabelecidas, obtendo assim sinais de áudio com variações de duração entre 3 e 10 segundos. A qualidade da gravação foi definida em sons do tipo mono e com 22khz de taxa de amostragem. Nesta configuração, o descarte de um sinal de áudio gravado ocorria sempre que um ruído externo perceptível se fazia presente ou quando o locutor, no ato da gravação, cometesse algum equívoco evidente como, por exemplo, a demora no início da


Figura 20 – Página do aplicativo *Voice Recorder* retirada da *Google Play Store*.

pronúncia ou na finalização da gravação, ou mesmo em erros de pronúncia da frase alvo. A Figura 21 traz uma visão quantitativa de áudios gravados em cada uma das sessões definidas para este projeto.

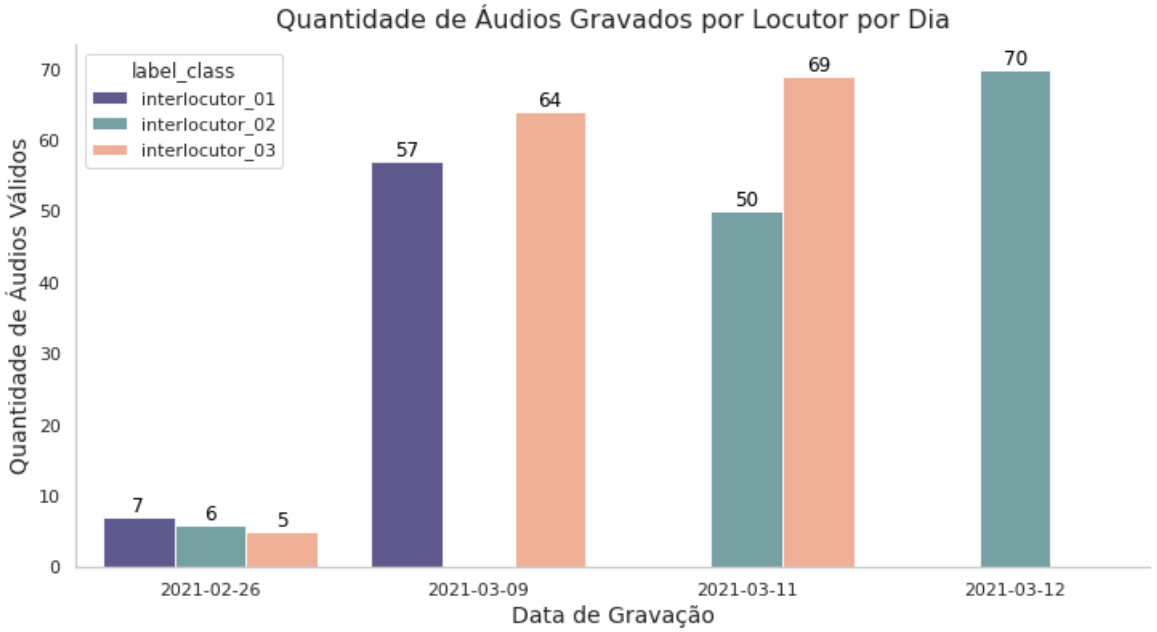


Figura 21 – Quantidade de áudios válidos registrados por cada locutor ao longo dos diferentes dias de gravações.

Em paralelo ao processo acima detalhado, um medidor de decibéis foi posicionado em um lugar neutro no ambiente de modo a registrar o nível sonoro obtido durante a etapa de gravação. Tal registro se deu via dispositivo móvel auxiliar ao dispositivo utilizado

na gravação, e nele foi instalado o aplicativo *Decibelímetro (Sound Meter)* (37) também disponível para sistemas operacionais Android via *Google Play Store*. Com isso, foi possível reunir insumos técnicos para a obtenção de sinais de áudio dentro de um parâmetro de qualidade definido, garantindo assim uma estabilidade no processo no que tange o objetivo de geração de registros gravados em ambientes silenciosos. A Figura 22 traz uma captura de tela da página do aplicativo *Decibelímetro (Sound Meter)*.

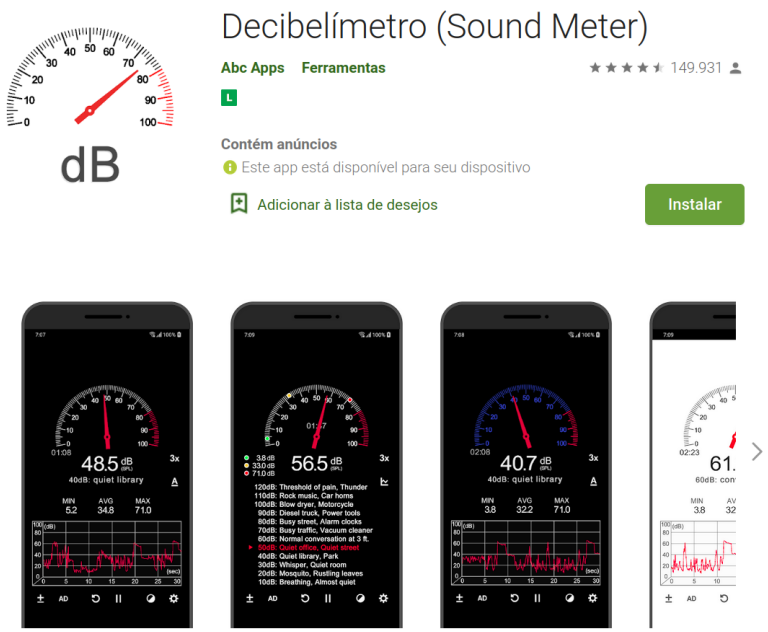


Figura 22 – Página do aplicativo *Decibelímetro (Sound Meter)* retirada da *Google Play Store*.

De forma ideal, os sinais eram considerados válidos quando na inexistência de ruídos externos perceptíveis e de grandes magnitudes durante a gravação, ou na obtenção de registros superiores ao limiar médio de 65 dB provenientes do decibelímetro, valor este que indica um ruído associado à conversa comum entre locutores (detalhes na Tabela 5).

Tabela 5 – Níveis sonoros em decibéis para algumas situações típicas.

| Fonte ou descrição do som | Nível de intensidade sonora, $\beta$ (dB) |
|---|---|
| Limiar da audição | 0 |
| Farfalhar de folhas | 10 |
| Sussurro médio | 20 |
| Rádio com volume baixo | 40 |
| Conversa comum | 65 |
| Tráfego pesado | 70 |
| Trem em um elevado | 90 |
| Show de rock | 110 |
| Limiear da dor | 120 |
| Turbina a jato | 140 |

Nesse contexto, a Figura 23 traz os dados do nível sonoro externo registrado pelo decibelímetro juntamente com valores quantitativos de áudios obtidos a cada sessão de gravação, e nela, é possível perceber níveis médio de ruído com variação entre 49 e 53 dB durante as sessões.

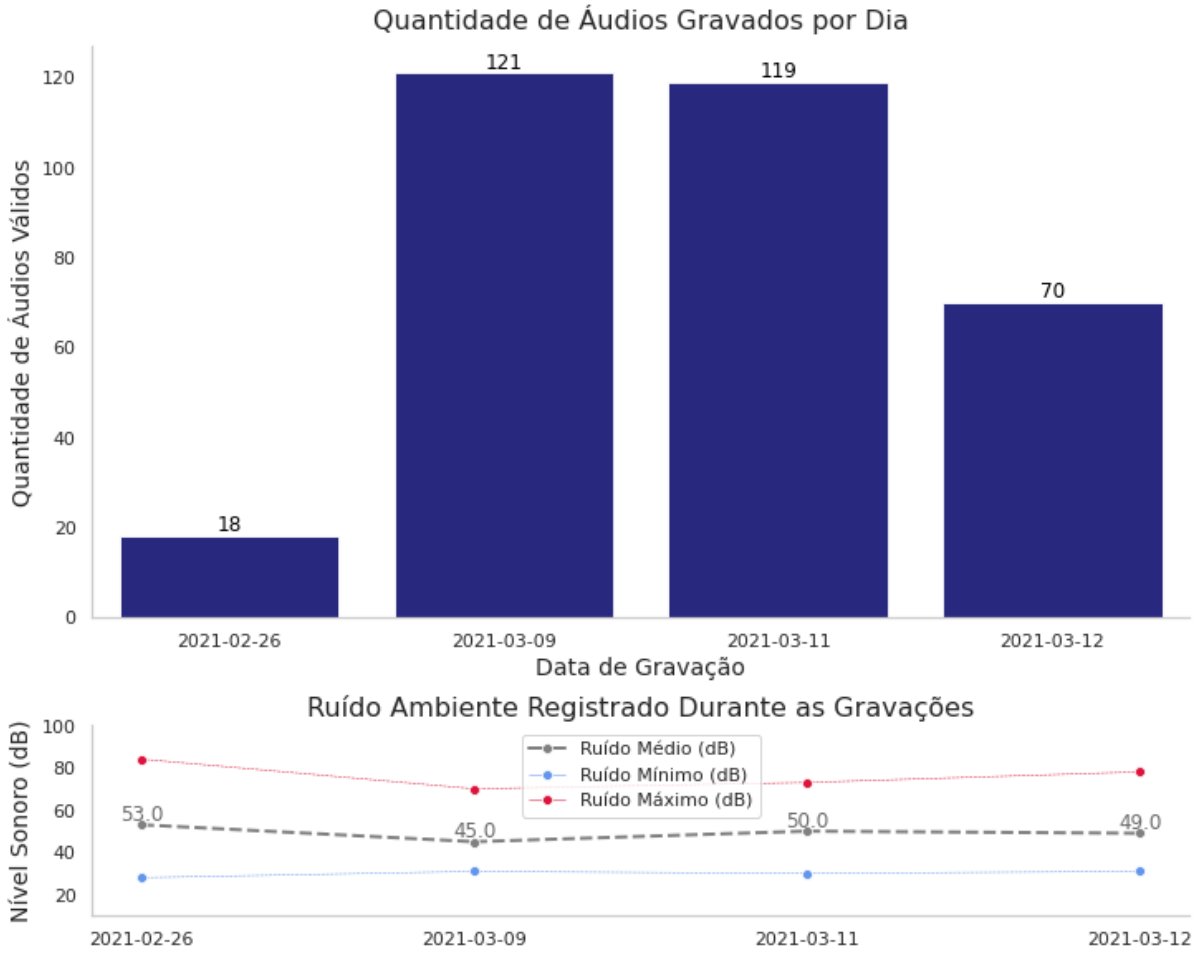


Figura 23 – Na faixa superior da figura, é mostrada a quantidade total de áudios gravados por dia (independente do locutor) e, na parte inferior, o nível sonoro de ruído registrado no ambiente em termos de média, mínimo e máximo em cada um dos dias de gravação.

Ao final das sessões individuais de gravação de cada um dos locutores, os arquivos mp3 resultantes eram transferidos do dispositivo móvel para um computador com o intuito de facilitar a identificação e a etiquetagem dos dados. O armazenamento foi definido a partir da criação de diretórios locais para guardar arquivos referentes a cada locutor envolvido no processo de gravação. A Figura 24 ilustra o diagrama completo do processo de gravação de sinais de áudio dentro do intuito deste trabalho.

Por fim, após a construção própria de uma base de dados seguindo um processo uniforme para todos os locutores envolvidos, ainda existia a necessidade de se obter dados para representar o que foi denominado neste trabalho como “locutores genéricos”. Com isso, o modelo criado poderá ser generalizado para o uso público, permitindo assim modelar

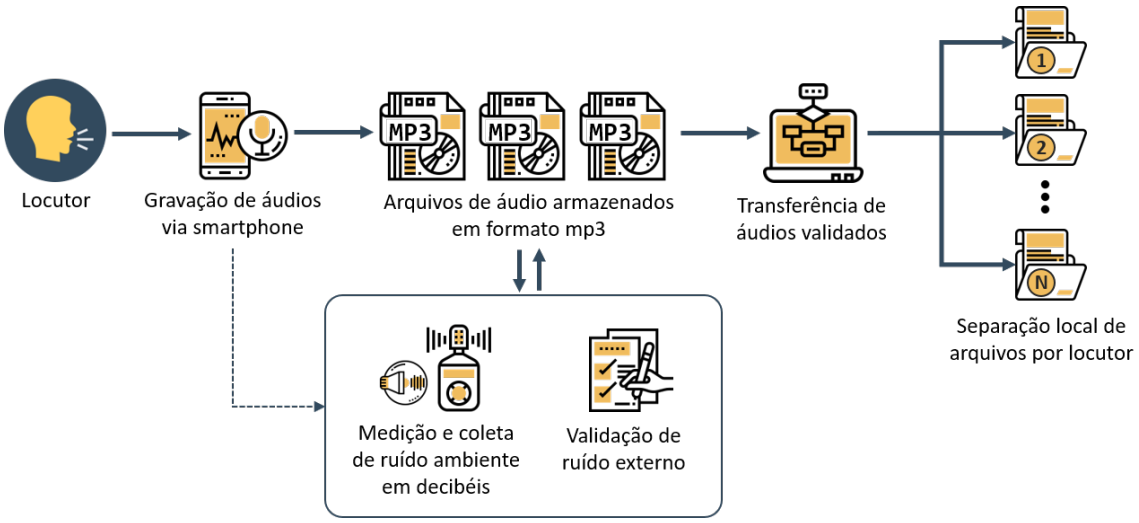


Figura 24 – Diagrama geral do processo de gravação de sinais de áudio para cada um dos locutores envolvidos como objeto de identificação do modelo de aprendizado final a ser treinado.

qualquer entrada de áudio recebida, não ficando restrito a apenas áudios de locutores que passaram pelo processo de gravação. E é nesse contexto que o próximo tópico descreve detalhes sobre o projeto conhecido como *Mozilla Common Voice*.

### 3.1.2 Mozilla Common Voice

Visando tornar o processo de identificação utilizável pelo público geral, fez-se necessário reunir insumos para treinar o modelo com uma resposta negativa a ser designada ao público solicitante cuja identidade não se faz presente na lista de locutores habilitados e que tiveram suas vozes coletadas e analisadas.

Desta forma, assim como mencionado em (7), dentro da possibilidade de existirem cenários onde a entrada de áudio a ser validada não está restrita aos indivíduos fornecedores de elementos de áudio para a modelagem do reconhecimento, é preciso considerar um modelo universal de *background*, ou UBM. A aplicação desse modelo caracteriza a construção de projetos de identificação de locutores em cenário aberto (ou *open-set scenario*).

Na literatura, a utilização de UBMs é relativamente comum, principalmente quando somada às GMMs. Diversos trabalhos relacionados (8, 13, 15, 24) alcançaram resultados expressivos dentro dessa combinação de modelos de reconhecimento de locutores conhecida por GMM-UBM. Considerando os objetivos deste trabalho de graduação e a dificuldade em encontrar modelos universais de locutores genéricos na língua portuguesa, decidiu-se adicionar elementos dessa categoria a partir da plataforma *Mozilla Common Voice* (16).

De forma direta, é possível definir o projeto *Common Voice* como uma iniciativa da Mozilla para ajudar a ensinar às máquinas como pessoas reais falam. A Figura 25 traz uma captura de tela da página principal do projeto e, nela, é possível identificar a possibilidade

de doar ou validar vozes. Através da difusão e ampla divulgação dessa frente, é possível obter sinais de áudio gravados e validados em diferentes idiomas, permitindo assim com que usuários interessados na utilização deste material possam desenvolver projetos altamente relevantes na área.

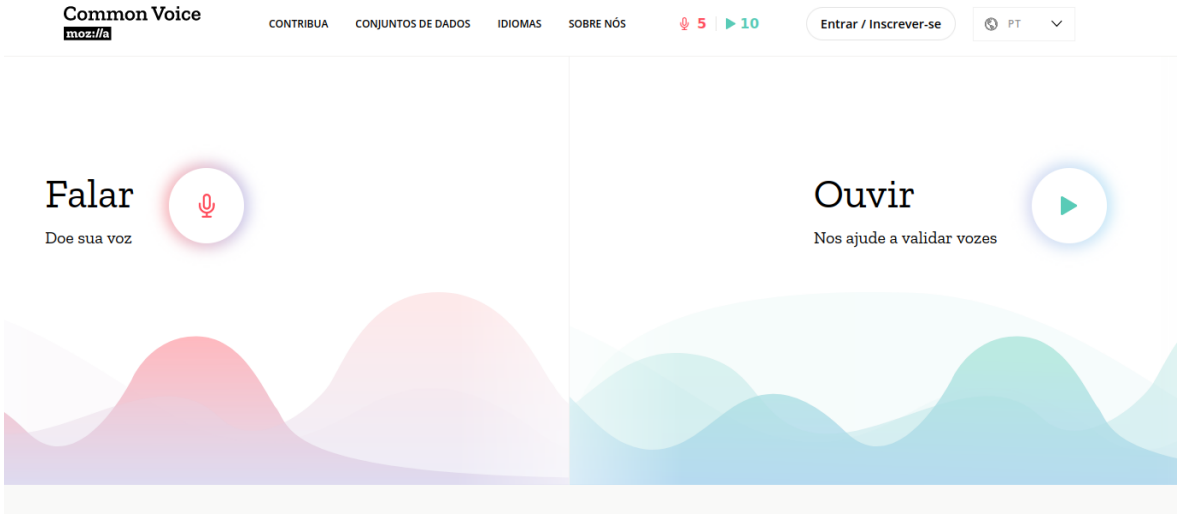


Figura 25 – Portal Mozilla Common Voice. À esquerda, usuários podem doar vozes a partir da leitura de frases curtas especificadas na tela. À direita, usuários podem validar vozes doadas para que estas possam ser utilizadas com um maior grau de confiança por desenvolvedores interessados.

O portal Common Voice, por fim, permite o *download* de uma base de dados contendo uma série de arquivos de áudios, validados e invalidados, para uso genérico, desde que seja assinado um termo de compromisso seguindo algumas diretrizes de boas práticas de utilização. A Figura 26 traz uma visão geral sobre o *corpus* disponível para áudios de língua portuguesa, possibilitando a análise dos metadados e da quantidade de elementos armazenados.

Assim, de acordo com os objetivos deste trabalho de graduação, os áudios de língua portuguesa reunidos a partir da base *Common Voice Corpus 6.1* serviram como insumo para a construção de um modelo de *background* universal próprio: com o devido acesso a áudios validados de locutores genéricos, pensou-se em utilizar esses elementos como uma classe adicional dentro do modelo supervisionado de aprendizado de máquina multiclasse. Em outras palavras, considerando um objetivo de identificação de locutores com $N$ classes distintas, cada uma representando um diferente locutor, os dados provenientes do corpus *Common Voice* seriam etiquetados como elementos da classe $N+1$, adicionando ao modelo de classificação multiclasse a possibilidade de classificar locutores genéricos, cujas vozes não foram registradas em nenhuma das outras $N$ classes.

Em uma adaptação da Figura 24, o processo de consolidação dos dados a serem utilizados na etapa de construção do modelo de reconhecimento de locutores, considerando

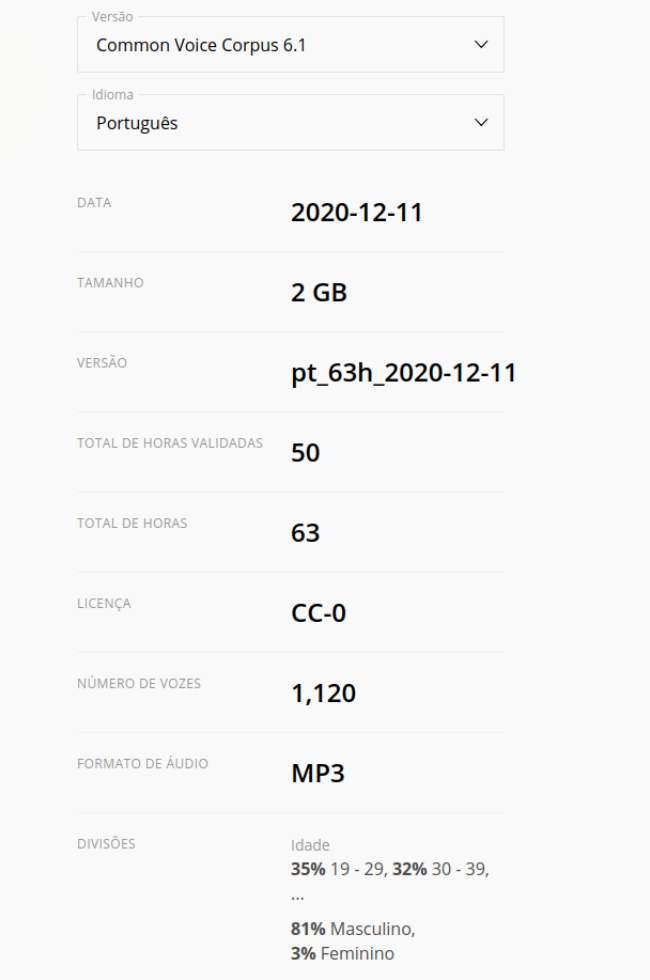


Figura 26 – Detalhes e metadados da base Common Voice Corpus 6.1 disponível em língua portuguesa e fornecida pelo portal Common Voice.

agora os elementos genéricos da base *Common Voice Corpus 6.1*, pode ser visualizado a partir da Figura 27.

Em resumo, a Tabela 6 define a terminologia a ser utilizada daqui em diante para os elementos representativos das classes pertecentes ao modelo de reconhecimento.

Tabela 6 – Classes utilizadas na construção do modelo de reconhecimento por voz.

| Referência | Descrição | *Array target (y)* |
|---|---|---|
| interlocutor 01 | Locutor a ser reconhecido na base com índice 1 | $[1, 0, 0, 0]$ |
| interlocutor 02 | Locutor a ser reconhecido na base com índice 2 | $[0, 1, 0, 0]$ |
| interlocutor 03 | Locutor a ser reconhecido na base com índice 3 | $[0, 0, 1, 0]$ |
| interlocutor 04 | Dados genéricos do *Common Voice* | $[0, 0, 0, 1]$ |

Uma vez centralizados os insumos, foi preciso então construir uma lógica capaz de realizar o gerenciamento e manuseio dos arquivos mp3 armazenados de modo a facilitar a leitura e a obtenção das características de áudio associadas. Na próxima seção, esse processo será detalhado e alguns exemplos dos resultados serão fornecidos.

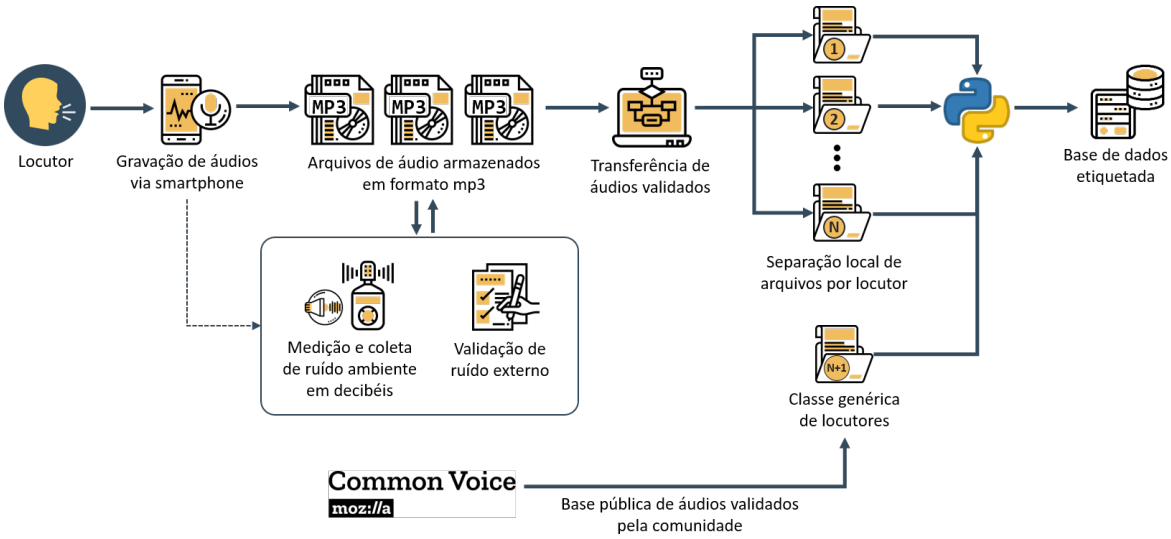


Figura 27 – Diagrama final de construção da base de dados contendo o fluxo de gravação de sinais de áudio e de enriquecimento de dados considerando a base pública Mozilla Common Voice.

### 3.1.3 Consolidação dos Áudios

Após a gravação de sinais de áudio de locutores a serem identificados no processo de reconhecimento automático e a obtenção de sinais de áudio de locutores genéricos consumidos pelo portal *Common Voice*, foi possível então construir uma lógica de armazenamento capaz de facilitar a leitura e a etiquetagem dos dados frente às classes definidas para a etapa de modelagem. Considerando a presença de três locutores protagonistas do reconhecimento por voz, foram criados três diretórios locais para o armazenamento dos respectivos áudios gravados pelos tais protagonistas. Adicionalmente, foi desenvolvido um código Python capaz de criar e enriquecer um diretório complementar para o armazenamento dos dados do *Common Voice*, gerando assim a classe genérica $N+1$ dentro do contexto exemplificado pela Tabela 6. A quantidade de áudios do *Common Voice* a ser utilizada no enriquecimento pode ser dinamicamente configurada através de um parâmetro definido no código Python desenvolvido.

Dessa forma, a Figura 28 traz o resultado obtido a partir da execução das funções *copy_common_voice()* e *read_data()*, sendo ambas construídas utilizando linguagem Python em conjunto com a biblioteca LibROSA (26) e responsáveis por entregar um objeto DataFrame do pandas (38) contendo metadados sobre os arquivos de áudio e as próprias séries temporais de áudios oferecidas como *arrays* de amplitude no domínio do tempo.

As variáveis presentes no DataFrame consolidado dos sinais de áudio são:

- **audio_path**: caminho completo do arquivo local armazenado na máquina;
- **filename**: referência do nome do arquivo de áudio armazenado;
- **file_format**: formato/extensão do arquivo de áudio;

```
# Copiando arquivos Mozilla Common Voice
if COPY_MCV:
    copy_common_voice(mcv_train_path=MCV_TRAIN_PATH, mcv_clips_path=MCV_CLIPS_PATH, data_path=AUDIOS_PATH,
                      n_mcv_files=N_MCV_FILES, label_prefix=LABEL_PREFIX, new_folder=NEW_FOLDER)

# Lendo áudios oficiais e gerando base inicial
df = read_data(data_path=AUDIOS_PATH, sr=SAMPLE_RATE, signal_col=SIGNAL_COL, target_col=TARGET_COL)
print(f'Dimensões da base inicial: {df.shape}')
df.head()
```

executed in 2m 11s, finished 20:30:39 2021-03-25

```
Dimensões da base inicial: (528, 7)
```

| | audio_path | filename | file_format | signal | duration | label_class | y_class |
|---|---|---|---|---|---|---|---|
| 0 | /home/paninit/workspaces/voice-unlocker/data/a... | 2021_03_11_21_32_48.mp3 | mp3 | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 4.310204 | interlocutor_03 | 3 |
| 1 | /home/paninit/workspaces/voice-unlocker/data/a... | 2021_03_11_21_31_43.mp3 | mp3 | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 4.310204 | interlocutor_03 | 3 |
| 2 | /home/paninit/workspaces/voice-unlocker/data/a... | 2021_03_11_21_31_52.mp3 | mp3 | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 4.075102 | interlocutor_03 | 3 |
| 3 | /home/paninit/workspaces/voice-unlocker/data/a... | 2021_03_11_21_10_17.mp3 | mp3 | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 11.859592 | interlocutor_03 | 3 |
| 4 | /home/paninit/workspaces/voice-unlocker/data/a... | 2021_03_09_21_04_36.mp3 | mp3 | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 7.497143 | interlocutor_03 | 3 |

Figura 28 – Base de dados transformada em um objeto DataFrame do pandas a partir da execução da função responsável pela leitura, consolidação e enriquecimento de metadados dos sinais de áudio gravados e disponíveis em diretórios locais.

- **signal**: sinal de áudio lido pela biblioteca LibROSA no domínio do tempo;
- **duration**: duração de cada um dos sinais da base;
- **label_class**: respectiva classe atrelada ao sinal de áudio;
- **y_class**: índice da classe em formato numérico.

Com isso, foi possível utilizar o DataFrame consolidado para a extração de informações adicionais sobre a base de dados construída. A Figura 29 traz uma visão de volumetria relacionada à quantidade total de arquivos de áudio gravados (ou obtidos, no caso do *Common Voice*) para cada classe de reconhecimento a ser utilizada na modelagem.

Ao todo, considerando os 528 arquivos que compõem a base, foram registrados aproximadamente 50 minutos de áudio gravados ou obtidos para a construção da base de dados. A Figura 30 traz uma relação do tempo total associado a cada locutor.

Por fim, considerando a execução das funções definidas para o gerenciamento e a leitura dos arquivos de áudio, todos os requisitos para a extração das características foram cumpridos. Dessa forma, a próxima seção trará um rico leque de exemplos práticos associados à extração dos atributos de áudio definidos dentro dos objetivos deste trabalho de graduação.

## 3.2 Extração de Características dos Sinais

Fornecido o contexto de construção da base de dados, tem-se início o processo de extração de características de sinais de áudio a serem utilizadas como insumos para o

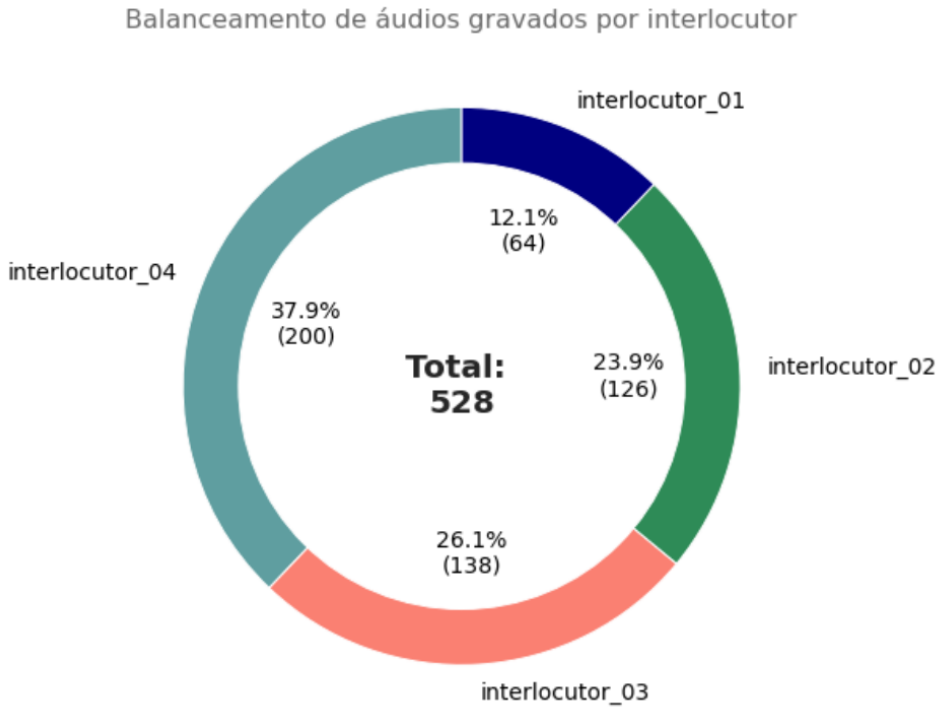


Figura 29 – Quantidade total de áudios gerados ou arquivos obtidos para cada um dos locutores presentes na base.

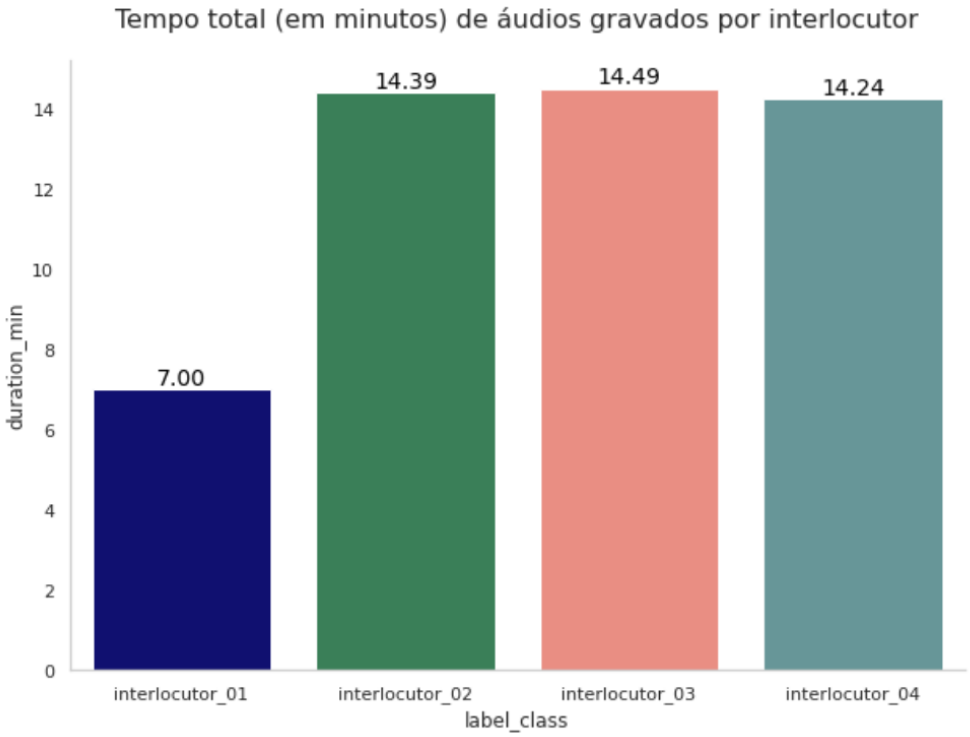


Figura 30 – Minutos totais considerados para cada locutor na construção da base.

treinamento de modelos de aprendizado capazes de identificar locutores a partir de suas respectivas vozes.

Essencialmente, assim como a variável idade pode ser um indicador relevante em

um contexto de modelagem de crédito, foram realizadas consultas na literatura para levantar algumas variáveis candidatas dentro do objetivo deste trabalho. Em (21), é possível encontrar uma série de características de sinais de áudio a serem analisadas para as mais diversas aplicações. Em (7) e em (24), é possível obter um detalhamento maior sobre as principais características de áudios voltadas exclusivamente para o contexto de reconhecimento de locutores. Analisando cenários práticos, como os exemplificados em (8), (4) e em (5), notou-se uma grande tendência na aplicação dos MFCCs como forma de extração de significados de áudios.

Considerando uma abordagem explorativa dentro do desenvolvimento próprio de um modelo de reconhecimento de locutores, a Tabela 7 mostra todas as características de sinais de áudios consideradas dentro da etapa de preparação dos dados para este projeto. Como principal motivação, a escolha de tais características visa propor um melhor entendimento sobre como os áudios podem ser caracterizados nos mais variados domínios, permitindo assim uma análise mais robusta e intuitiva do resultado final.

Tabela 7 – Características de sinais de áudio extraídas no *pipeline* de pré-processamento dos dados.

| Domínio | Característica |
|---|---|
| Tempo | Envelope de Amplitude |
| Tempo | Raíz da Energia Média Quadrática |
| Tempo | Zero Crossing Rate |
| Frequência | Taxa de Energia de Banda |
| Frequência | Centroide Espectral |
| Frequência | Largura de Banda |
| Tempo-Frequência | Espectrograma |
| Tempo-Frequência | MFCCs |

Obtendo, como ponto de início, o DataFrame consolidado com os sinais de áudio exemplificado pela Figura 28, foram escolhidos, de forma aleatória, três sinais de áudio referentes a cada um dos locutores protagonistas de gravação para que, no decorrer da extração das características dos sinais, fosse possível visualizar exemplos práticos do significado de cada variável extraída. Desta forma, a Figura 31 traz uma análise dos sinais selecionados dos locutores 1, 2 e 3 no domínio do tempo em gráficos separados e, através da Figura 32, é possível visualizar os sinais sobrepostos no tempo.

Já neste momento, é possível perceber algumas diferenças associadas à amplitude de voz de cada locutor ou mesmo atreladas a algum viés de gravação em termos de inicialização e finalização. Porém, trata-se de apenas uma única amostra de cada locutor, sendo impossível concluir diferenciações mais aprofundadas entre os protagonistas da base. Neste ponto, serão propostas construções de códigos para extração e visualização das características detalhadas pela Tabela 7 obtendo, como referência, os três sinais

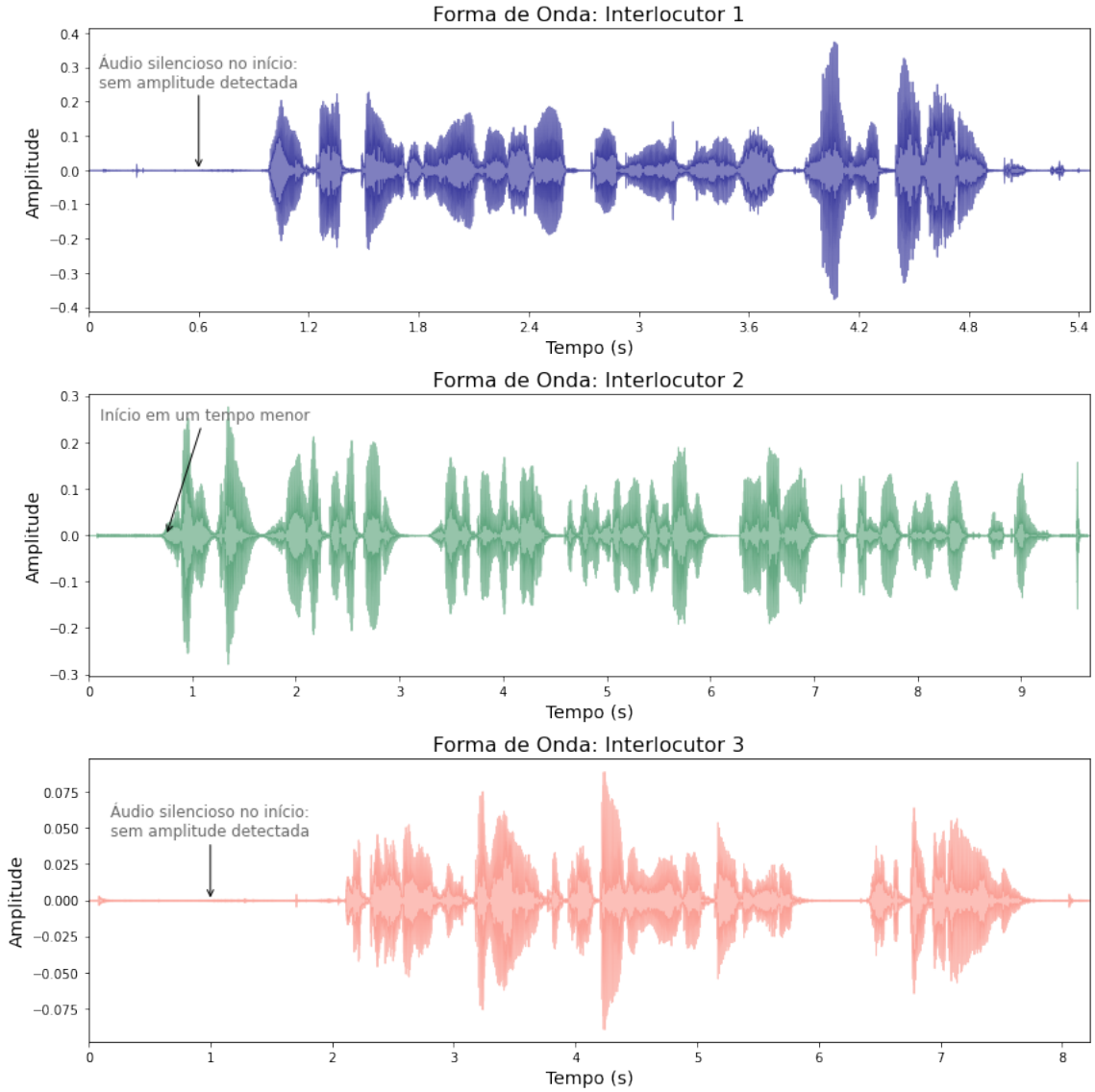


Figura 31 – Exemplo de sinais de áudio no domínio do tempo para cada um dos três locutores progatonistas da gravação.

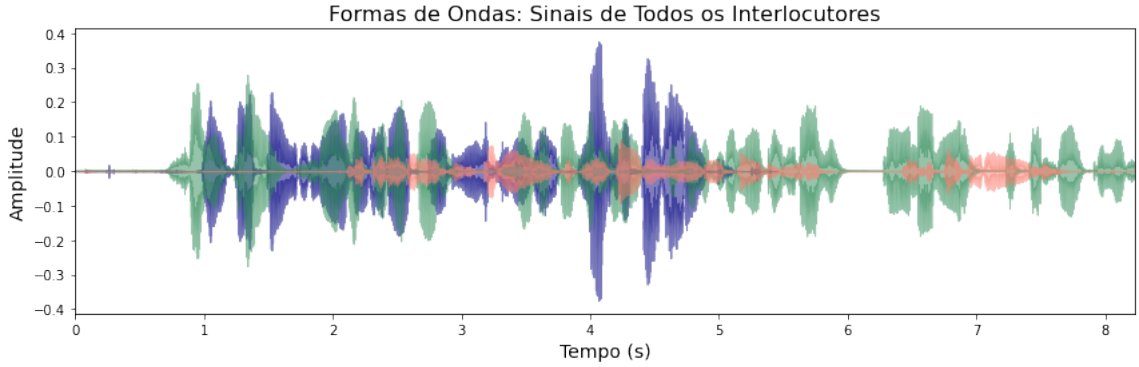


Figura 32 – Sinais de áudio para cada um dos três locutores posicionados em um mesmo eixo de plotagem para feitos comparativos.

exemplificados pelas Figuras 31 e 32. Como variáveis de projeto, foi definido o uso de um

*frame* contendo 1024 amostras e um deslocamento lateral de 512 amostras entre frames.

O cálculo do envelope de amplitude, dado a partir da equação (2.7), foi aplicado aos sinais de exemplo, resultando assim no contorno vermelho ilustrado pela Figura 33.

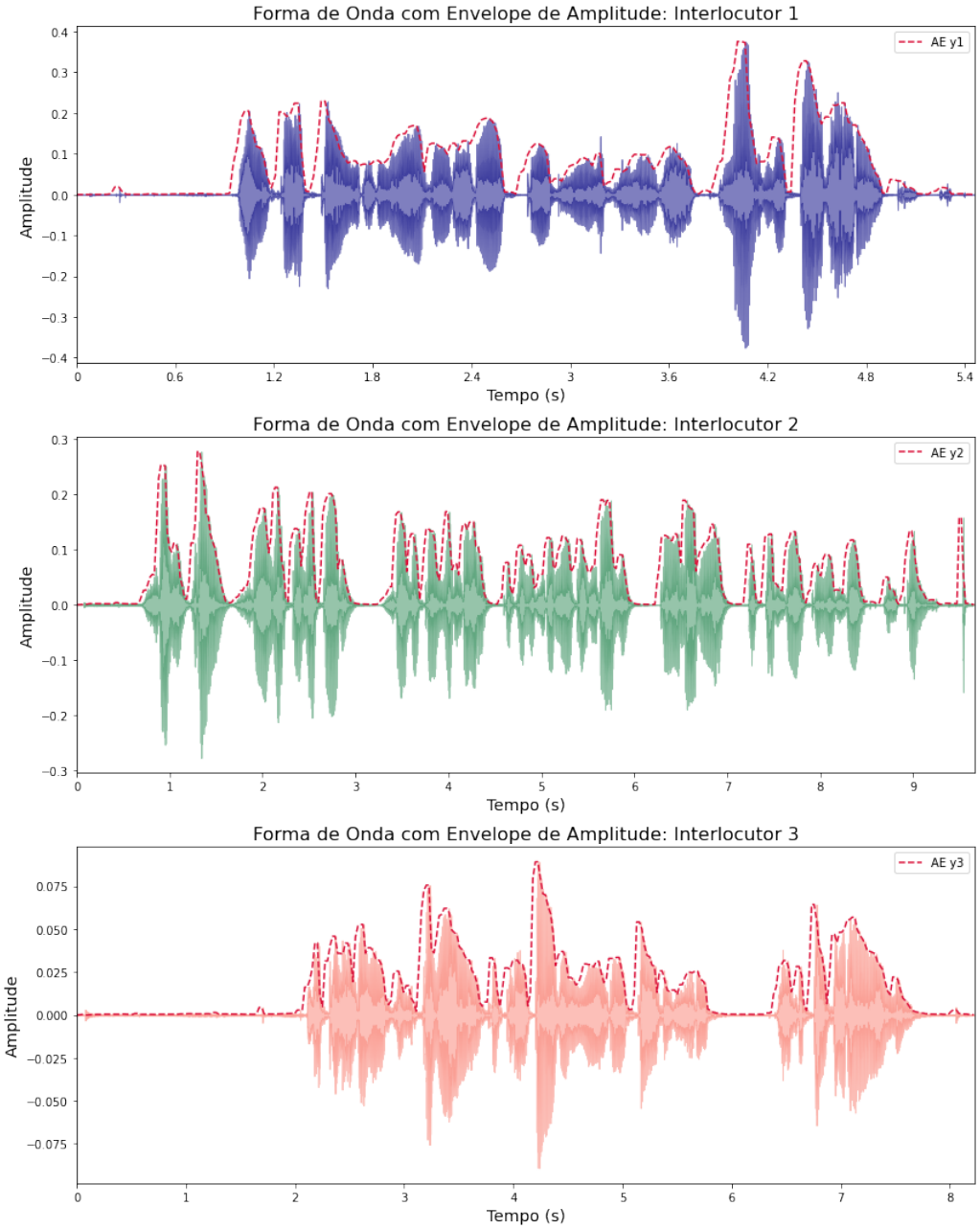


Figura 33 – Envelope de amplitude aplicado aos sinais de exemplo.

Na sequência, a raíz da energia média quadrática (energia RMS) de cada um dos sinais foi extraída a partir da equação (2.8) junto a métodos disponíveis na biblioteca LibROSA. Seu resultado visual é dado pela Figura 34.

Encerrando o conjunto de características extraídas no domínio do tempo, foi aplicado aos sinais o *zero crossing rate*, sendo este obtido através da aplicação da equação (2.9) e em método também disponível pela biblioteca LibROSA. O resultado para cada

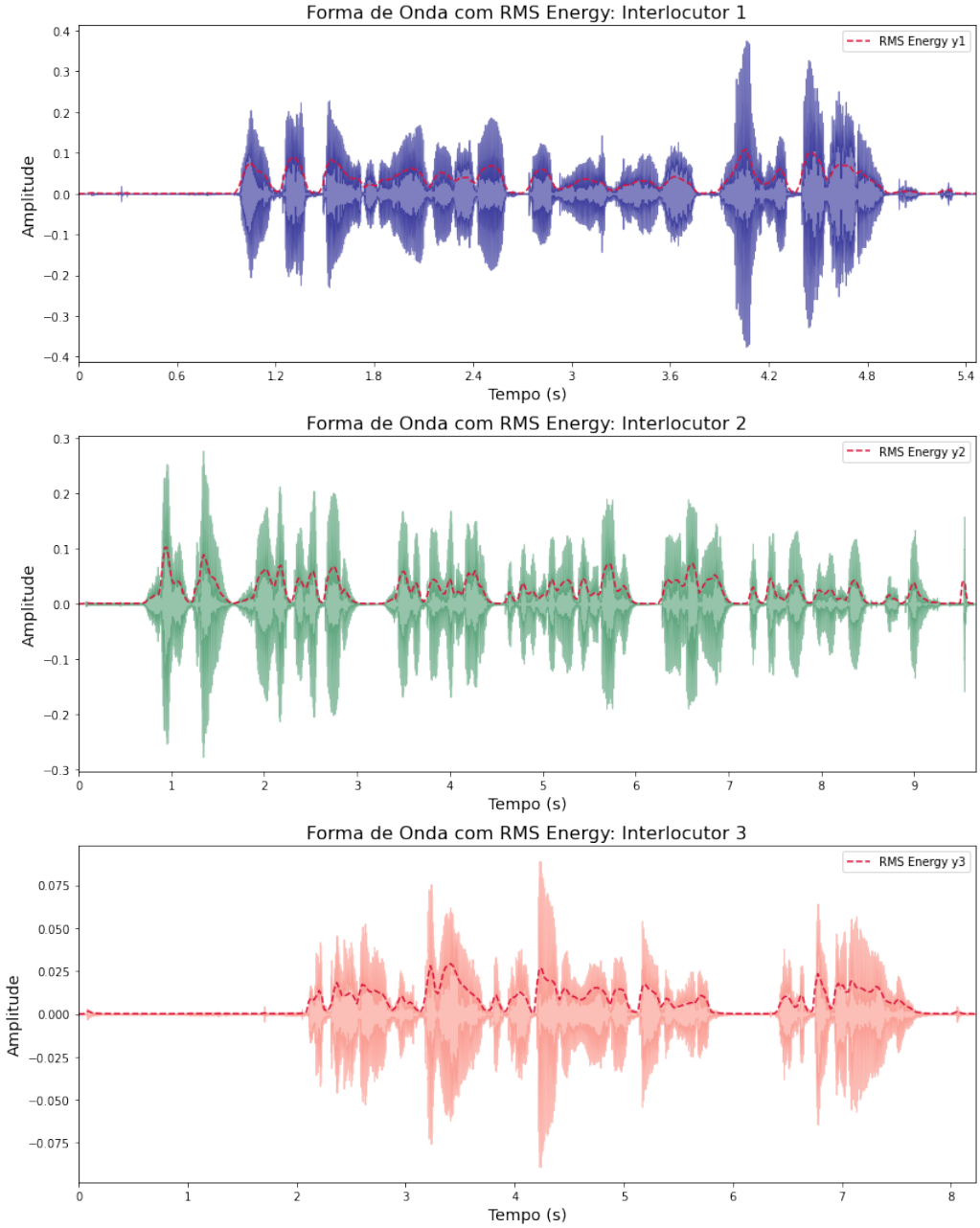


Figura 34 – Raíz da energia média quadrática aplicada aos sinais.

locutor pode ser visualizado a partir da Figura 35.

Introduzindo as características dos sinais no domínio da frequência, a Figura 36 demonstra o espectro de frequências obtido para cada sinal de exemplo a partir da aplicação da transformada de Fourier. Essa visão é essencial dentro da extração de características neste domínio, visto que, a partir dela, é possível relacionar os elementos que compõem o sinal e que podem ser utilizados, de alguma forma, em etapas de modelagem. O cálculo da transformada de Fourier foi obtido através do uso de funcionalidades da biblioteca *numpy* (39), mais especificamente de seu módulo *fft*.

Assim, a taxa de energia de banda (ou BER) de cada sinal de exemplo, sendo

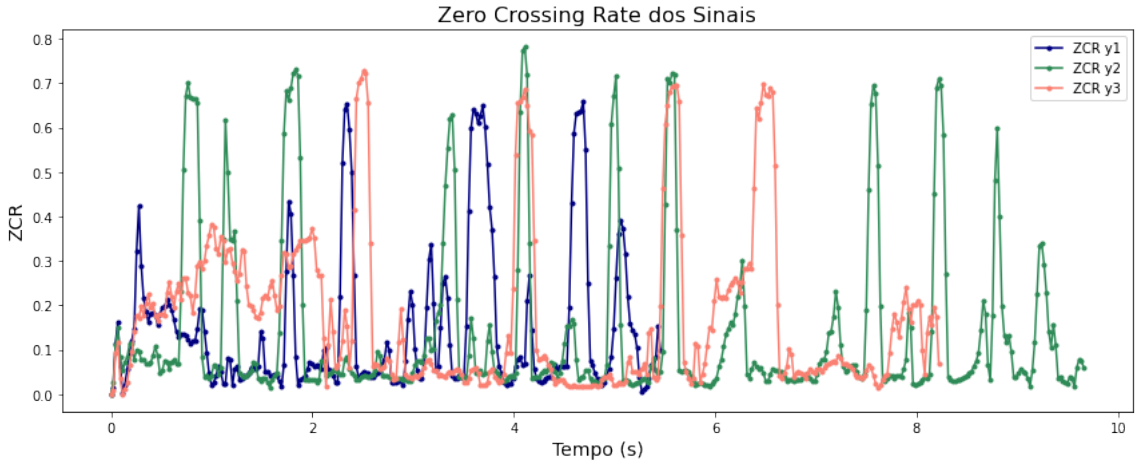


Figura 35 – Zero crossing rate aplicado aos sinais de exemplo.

esta definida pela equação (2.15), foi calculada e seu resultado prático pode ser observado através da Figura 37.

Na sequência, foi extraído o centroide espectral dos sinais associados aos diferentes locutores. Seu cálculo é definido pela equação (2.16) e, na prática, a biblioteca LibROSA possui implementações próprias a partir de seus métodos internos. O resultado da extração do centroide espectral pode ser visualizado na Figura 38.

Por fim, encerrando a etapa de extrações de características no domínio da frequência, foi proposta a análise da largura de banda, sendo esta definida pela equação (2.17). Visualmente, é possível observar a largura de banda de cada um dos sinais através da Figura 39.

Em seguida, como ponto de entrada para a análise de características de sinais de áudio no domínio tempo-frequência, a aplicação da STFT se fez presente para que fosse possível adicionar a dimensão temporal no espectro de frequências puramente obtido pela FFT. Assim, utilizando a equação (2.14) em conjunto com módulos presentes na biblioteca LibROSA, foi possível calcular e traçar espectrogramas para cada um dos sinais de exemplo, proporcionando assim o resultado exemplificado pela Figura 40.

De modo a propor uma análise mais detalhada, a Figura 41 traz uma comparação entre os espectrogramas gerados para cada um dos sinais e suas respectivas representações na escala Mel. A base de cálculo utiliza a equação (2.18) para a conversão de elementos de frequência, originalmente dados em Hertz para a escala Mel. Além disso, utilizou-se ferramentas da biblioteca LibROSA para a geração da matriz espectral na escala Mel considerando um número de coeficientes (ou quantidade de bandas Mel) igual a 90. Na imagem, é possível observar o destaque maior dado para componentes de baixa frequência dentro da escala Mel dentro da perspectiva de percepção humana de sons.

Assim, analisado o contexto e a importância da análise de espectrogramas na

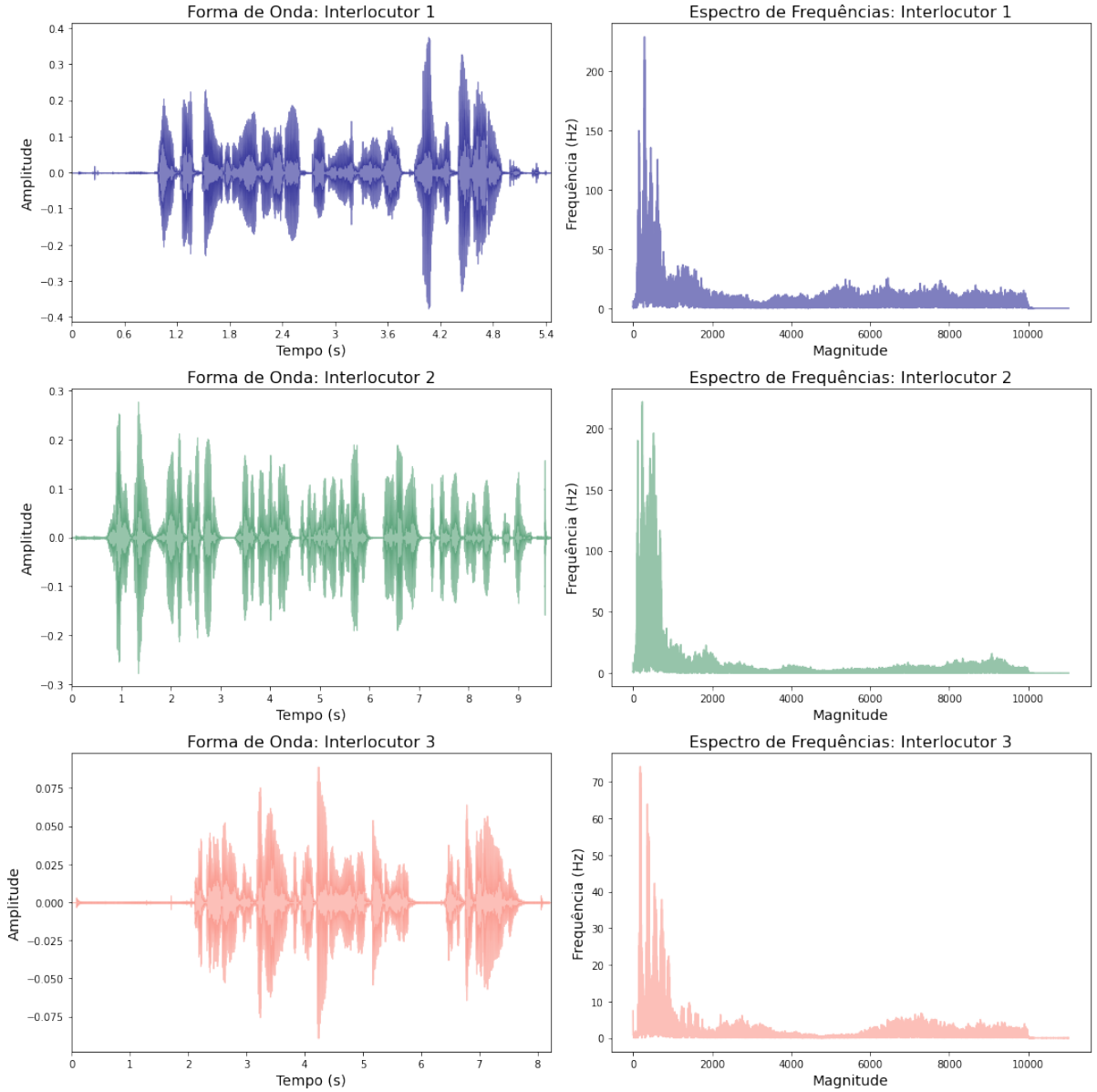


Figura 36 – Espectro de frequências gerado para cada sinal de exemplo através da FFT.

escala Mel, foi desenvolvido um código capaz de calcular e traçar as *Mel-frequency cepstral coefficients* para cada um dos sinais de exemplo. Como mencionado anteriormente, os MFCCs são extremamente relevantes dentro do contexto de caracterização de sinais de áudio, visto que sua obtenção é dada a partir de fundamentos que consideram a percepção humana e a geração de fonemas fundamentais dentro da construção da fala. A biblioteca LibROSA fornece métodos para a extração das MFCCs de sinais de áudio dispostos no domínio do tempo e, dentro da proposta desta seção, foi desenvolvido um código que, além de realizar a extração dos coeficientes, calcula também suas respectivas primeira e segunda derivadas, gerando assim uma visão espectral desses três universos de análise representados pela Figura 42. O cálculo tem como base a utilização dos 13 primeiros coeficientes MFCCs dos sinais.

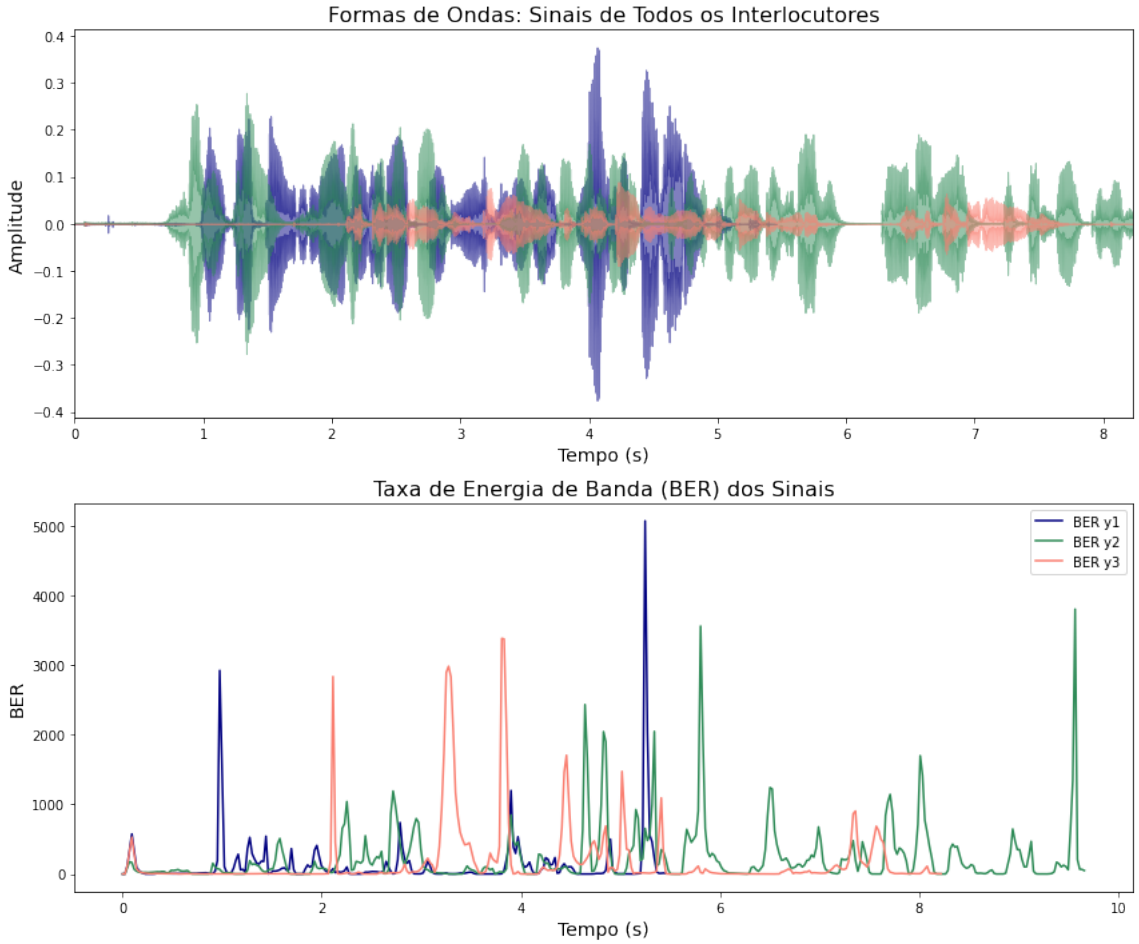


Figura 37 – Taxa de energia de banda calculada para cada um dos sinais de exemplo. Na parte superior da imagem, os sinais são representados no domínio do tempo. Na parte inferior, tem-se a taxa de energia de banda para cada frame associado ao sinal.

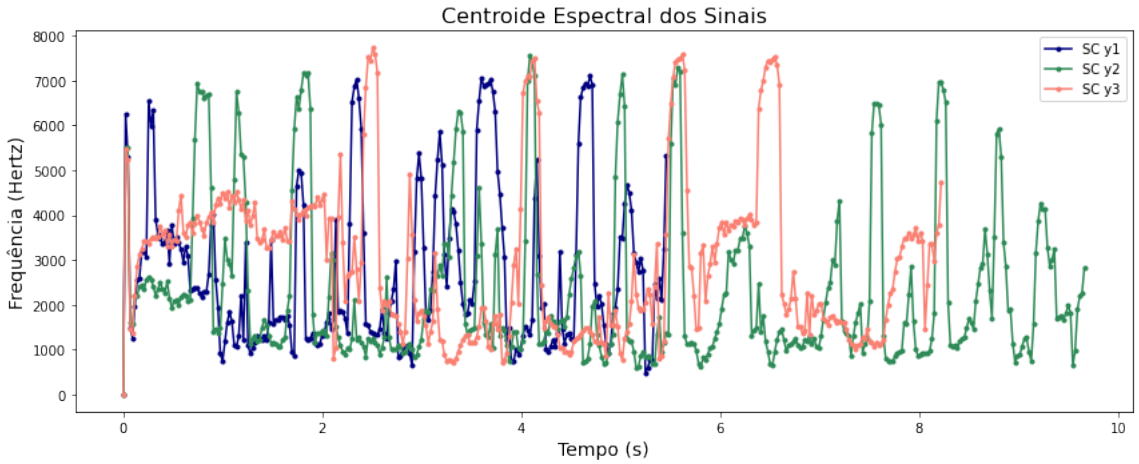


Figura 38 – Centroide espectral extraído de cada um dos sinais de exemplo.

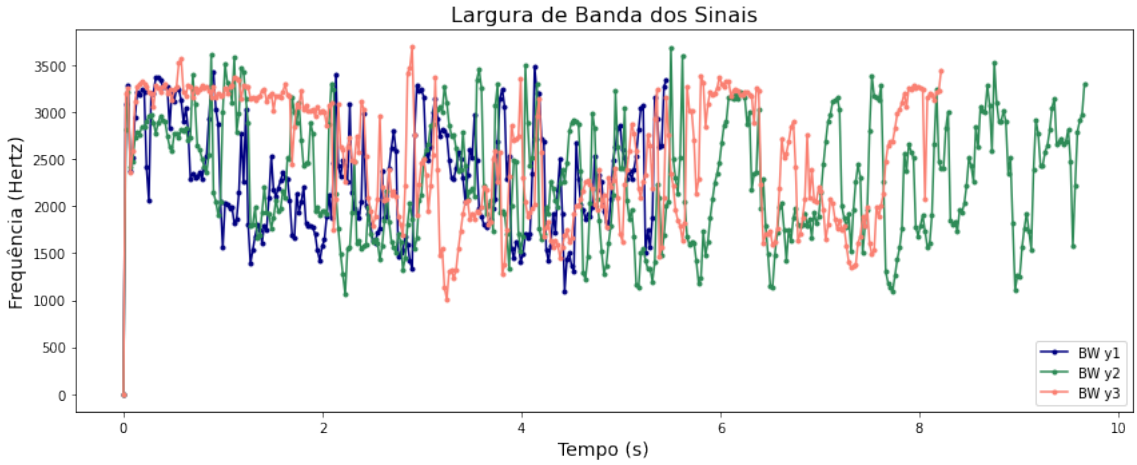


Figura 39 – Largura de banda extraída de cada um dos sinais de exemplo.

## 3.3 Pipeline de Preparação dos Dados

Após uma longa jornada na exploração e extração de características de sinais de áudio nos domínios do tempo, frequência e tempo-frequência, a próxima etapa dentro do contexto de modelagem tem como base a construção de um *pipeline* ou fluxo de preparação dos dados. Neste cenário, diferente de uma proposta visual de análise das características, o objetivo gira em torno da construção de um código estruturante capaz de receber sinais de áudio como entrada e retornar as características agregadas em um formato matricial a ser utilizado como entrada para modelos de aprendizado de máquina.

Na seção anterior, foi possível notar que os resultados obtidos para cada característica definida eram dados a partir de *arrays* temporais com relação direta à quantidade de amostras definidas para cada *frame* do sinal. Em termos de modelagem, é necessário adicionar um passo relacionado à agregação estatística desses resultados de modo a propor características únicas para cada sinal de entrada. Em outras palavras, para a construção da base final a ser utilizada como insumo dos modelos preditivos, propõe-se a aplicação de agregações estatísticas nos *arrays* gerados a partir das extrações das características para cada *frame* do sinal. Dessa forma, levando em consideração o caráter explorativo e analítico deste projeto, decidiu-se aplicar, inicialmente, os seguintes agregados estatísticos:

- Média
- Mediana
- Desvio Padrão
- Variância
- Máximo
- Curtose
- Assimetria

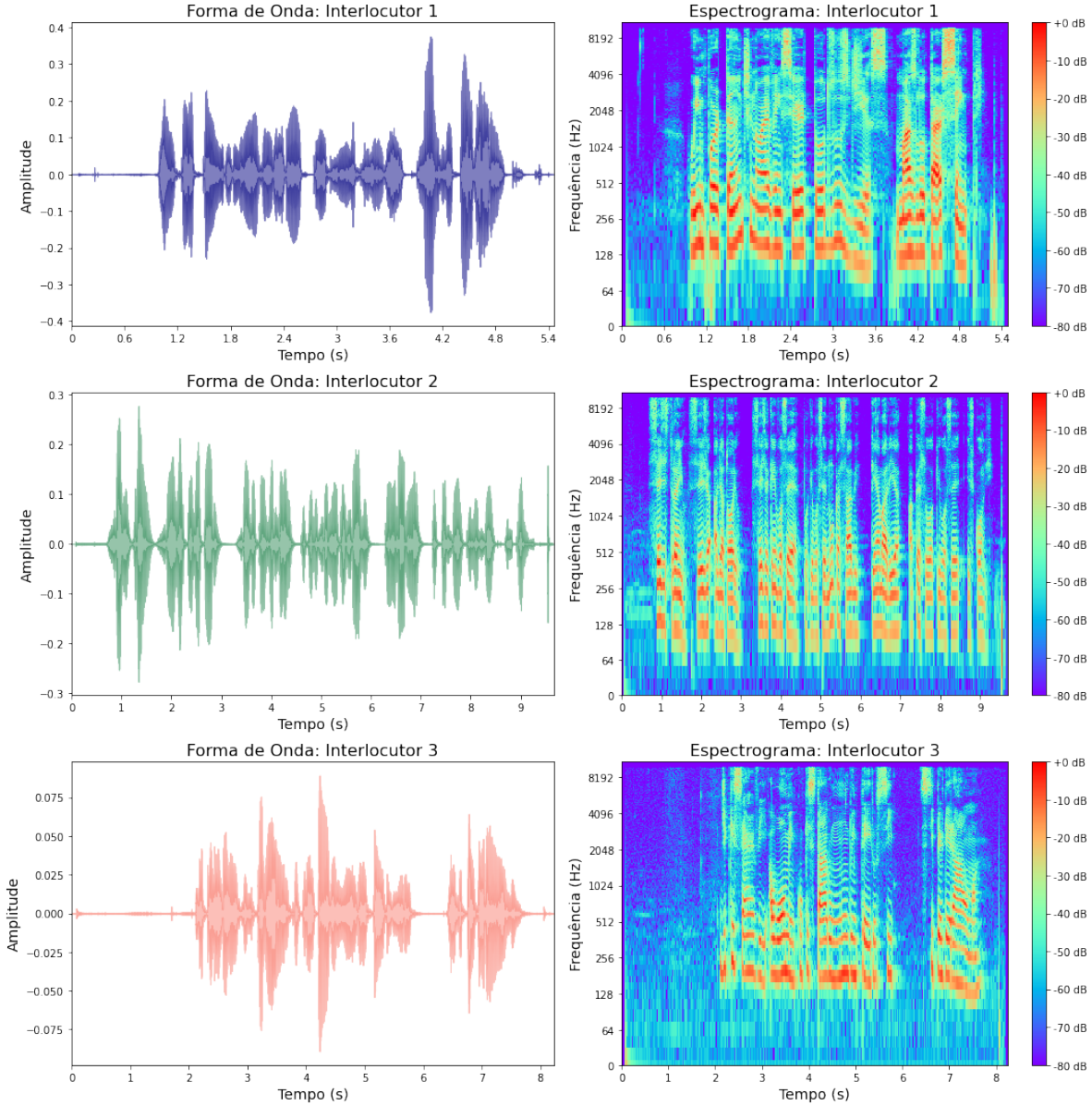


Figura 40 – Espectrogramas traçados para cada um dos sinais de exemplo.

De modo a construir um fluxo sólido de preparação que pudesse ser facilmente reproduzido ou adaptado, utilizou-se as funcionalidades do módulo *Pipeline* da biblioteca *sklearn* (29). A partir desse módulo, é possível definir etapas ou passos de transformação a serem executados sequencialmente, permitindo assim a execução de diversas transformações em uma base de entrada a partir do acionamento de um único elemento. Em (27), é possível encontrar exemplos práticos relacionados à construção de *pipelines* de transformação atrelados às mais variadas etapas.

Assim, cada transformador dentro deste fluxo de preparação proposto deve ser construído a partir de uma classe em Python que necessariamente deve herdar as classes *BaseEstimator* e *TransformerMixin* também importadas da biblioteca *sklearn*. Essa nova classe transformadora contém, opcionalmente, um método construtor (definido pela palavra

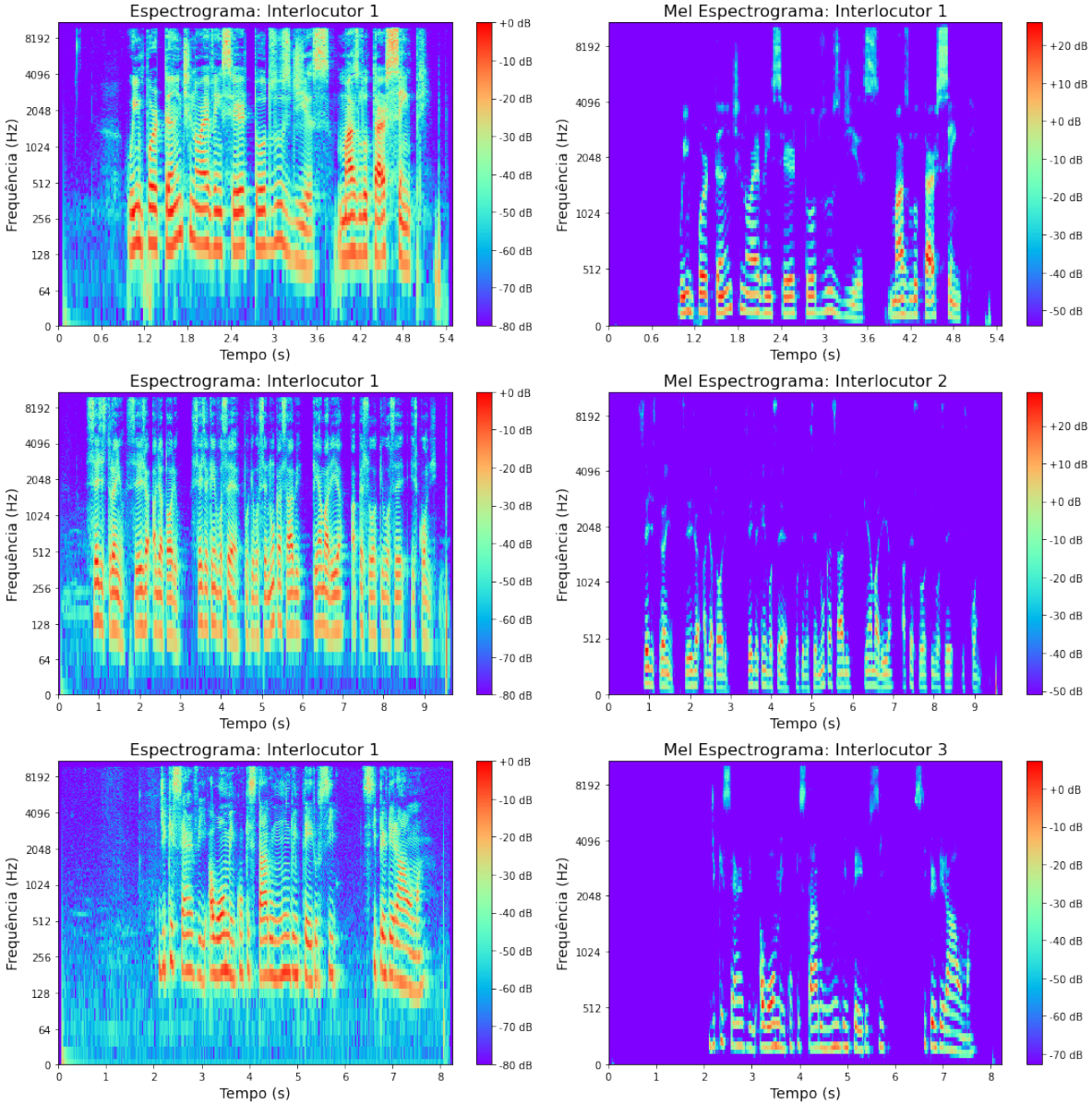


Figura 41 – Espectrogramas traçados para cada um dos sinais de exemplo.

reservada *___init___*) e, necessariamente, os métodos *fit()* e *transform()*. Ao herdar, em sua definição, as classes *BaseEstimator* e *TransformerMixin*, a classe transformadora automaticamente herda o método *fit_transform()*, sendo este último utilizado para a devida preparação e execução do código de transformação dos dados. Para ilustrar esse entendimento, a Figura 43 traz a aplicação da classe *AmplitudeEnvelop* construída especialmente com o objetivo de extração agregada do envelope de amplitude dos sinais de entrada presentes na base consolidada dos áudios.

No código ilustrado pela Figura 43, o atributo da classe transformadora representado pela variável *FEATURE_AGGREG* contém uma lista de agregadores a serem aplicados na característica extraída do sinal. Essencialmente, esta variável pode ser modificada de acordo com os propósitos de análise da modelagem, bastando adicionar ou retirar

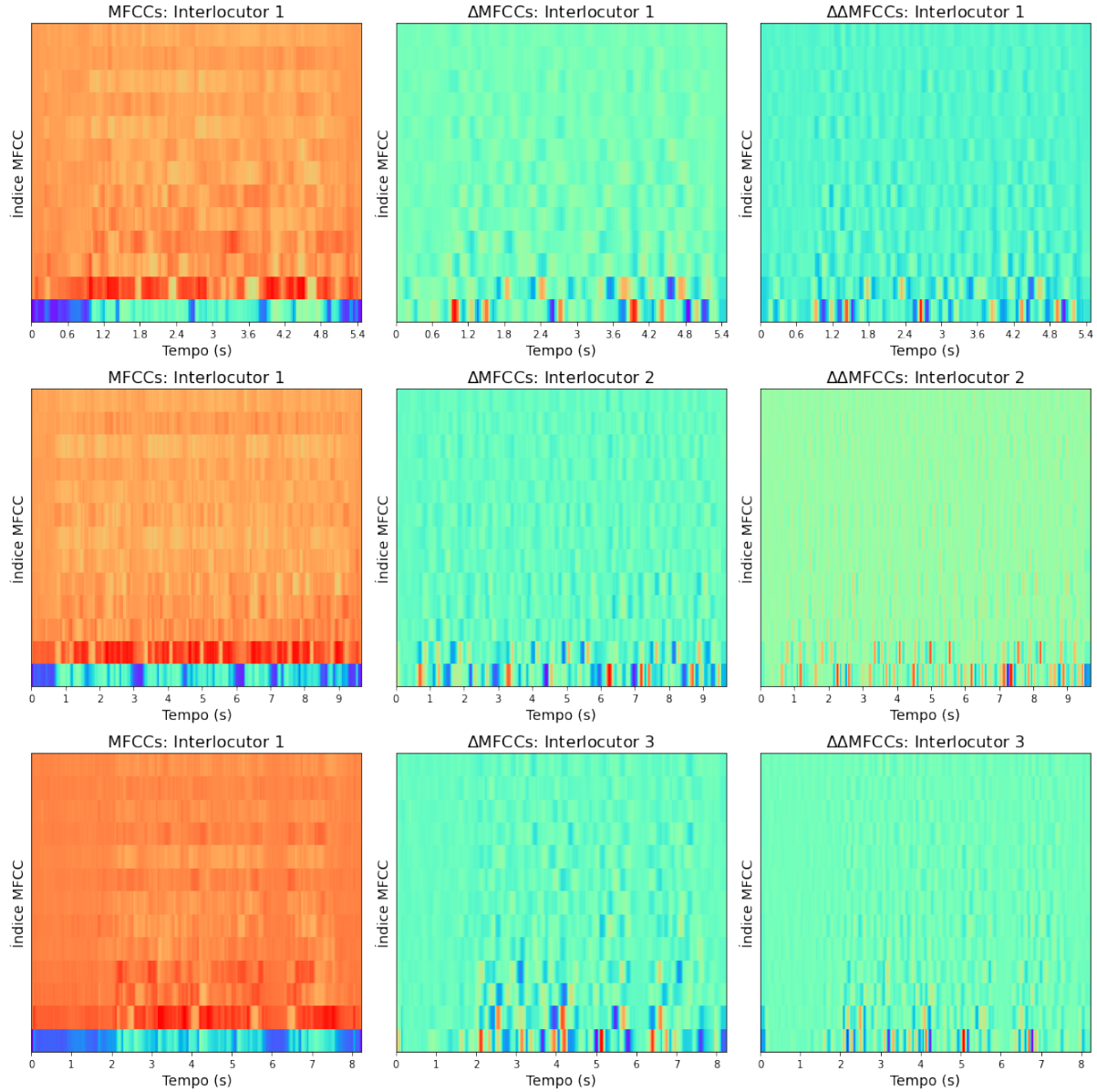


Figura 42 – MFCCs dos sinais de áudio de exemplo em três diferentes abordagens: extração pura, primeira derivada e segunda derivada.

elementos.

Portanto, levando em consideração a expansão da construção de classes transformadoras para cada uma das características definidas dentro dos objetivos deste trabalho, a Tabela 8 traz consigo uma forma de referenciar as nomenclaturas das classes em Python com seus respectivos objetivos de extração.

Ao todo, considerando todas as classes transformadoras detalhadas acima, a base final de modelagem é composta por 329 características distintas de sinais de áudio extraídas a partir dos agregados estatísticos definidos previamente. Em termos técnicos, espera-se que essa quantidade de características seja computacionalmente custosa, fato este que será analisado posteriormente na seção específica de construção do modelo de reconhecimento.

```
# Definindo transformador para envelope de amplitude
class AmplitudeEnvelop(BaseEstimator, TransformerMixin):
```

executed in 16ms, finished 11:35:31 2021-03-26

```
# Criando objeto e retornando features agregadas
ae_extractor = AmplitudeEnvelop(frame_size=FRAME_SIZE, hop_length=HOP_LENGTH,
                                signal_col='signal', feature_aggreg=FEATURE_AGGREG)
X_ae = ae_extractor.fit_transform(X)
X_ae.head()
```

executed in 16.2s, finished 11:35:47 2021-03-26

| | signal | ae_mean | ae_median | ae_std | ae_var | ae_max | ae_kurtosis | ae_skew |
|---|---|---|---|---|---|---|---|---|
| **0** | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 0.026926 | 0.024551 | 0.025405 | 0.000645 | 0.100281 | -0.407265 | 0.657465 |
| **1** | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 0.028720 | 0.021637 | 0.023985 | 0.000575 | 0.082886 | -0.722284 | 0.601592 |
| **2** | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 0.028412 | 0.018982 | 0.029994 | 0.000900 | 0.117493 | 0.216276 | 1.022808 |
| **3** | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 0.035016 | 0.030731 | 0.026640 | 0.000710 | 0.115021 | 0.366818 | 0.853834 |
| **4** | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ... | 0.046102 | 0.038208 | 0.043858 | 0.001923 | 0.208038 | 1.570043 | 1.269211 |

Figura 43 – Exemplo de classe transformadora construída em Python para a extração da característica envelope de amplitude já considerando os agregados estatísticos definidos.

Futuramente, pode-se estudar técnicas de redução de dimensionalidade ou do que pode ser chamado de seleção de atributos (também comumente conhecido por *feature selection*), proporcionando um menor custo computacional e uma possível melhora no desempenho final do modelo.

Por fim, a construção do fluxo completo de preparação envolve a utilização da classe *Pipeline* da biblioteca *sklearn* para a consolidação de todas as classes transformadoras desenvolvidas no processo. A Figura 44 traz o código construído para a geração do objeto *audio_fe_pipeline* responsável por aplicar, sequencialmente, todas as classes de transformação definidas previamente. O resultado é uma matriz multidimensional de dimensões (528, 329), sendo a primeira representando a quantidade de sinais de áudio gravados ou obtidos, e a segunda representando a quantidade total de características extraídas de cada um dos sinais.

Dessa forma, o objeto *audio_fe_pipeline* é salvo localmente na máquina em formato *pkl*, permitindo sua posterior leitura e aplicação através da biblioteca *joblib* disponível em Python.

## 3.4 Treinamento de Modelo de Identificação de Locutores

Conforme pontuado ao longo deste trabalho de graduação, o passo sequencial à extração de características de sinais de áudio e preparação analítica de uma base de dados capaz de armazenar tais atributos diz respeito à utilização de algoritmos de aprendizado de máquina em sua abordagem mais clássica, de modo a construir um modelo capaz de reconhecer locutores a partir de novos sinais de áudios fornecidos como entrada.

Tabela 8 – Classes transformadoras dentro do *pipeline* de transformação dos sinais de áudio.

| Classe | Objetivo | *Features* geradas |
|---|---|---|
| AmplitudeEnvelop() | Extração de agregados estatísticos do envelope de amplitude dos sinais | 7 |
| RMSEnergy() | Extração de agregados estatísticos da raíz quadrada da enérgia média | 7 |
| ZeroCrossingRate() | Extração de agregados estatísticos da taxa de cruzamento de eixo dos sinais | 7 |
| BandEnergyRatio() | Extração de agregados estatísticos da taxa de energia de banda dos sinais | 7 |
| SpectralCentroid() | Extração de agregados estatísticos do centroide espectral dos sinais | 7 |
| BandWidth() | Extração de agregados estatísticos da largura de banda dos sinais | 7 |
| GroupSpecAggreg() | Extração de agregados estatísticos da energia espectral dividida em altas e baixas frequências | 14 |
| MFCCsAggreg() | Extração de agregados estatísticos dos coeficientes MFCCs puros, primeira e segunda derivadas | 273 |

Assim, esta seção tem início logo após a construção do *pipeline* de preparação e extração de *features* dos sinais de áudio. Neste cenário, utilizou-se a biblioteca *pycomp* (40), de desenvolvimento próprio, para facilitar e otimizar todas as etapas atreladas à modelagem proposta.

Em linhas gerais, é possível afirmar que a biblioteca *pycomp* é uma ferramenta construída na camada superior de outras bibliotecas fundamentais em Python, como pandas, numpy, sklearn, matplotlib, entre outras. Seu principal objetivo é consolidar métodos, classes e funções úteis no uso cotidiano de programadores Python, permitindo a construção de aplicações complexas utilizando poucas linhas de código através da implementação de componentes previamente desenvolvidos para uso. Para o contexto de aprendizado de máquina, o pacote *pycomp* traz consigo o módulo *pycomp.ml.trainer* com algumas classes e funções extremamente úteis dentro deste cenário. Neste universo, é necessário citar que a classe *ClassificadorMulticlasse* foi desenvolvida como motivação aos propósitos deste trabalho de graduação, sendo esta adicionada às versões mais recentes da biblioteca *pycomp*.

Como primeiro passo para o ingresso na fase de modelagem, será proposta a separação do conjunto de dados preparado em três subconjuntos distintos: treino, validação e teste. Esta divisão é extremamente importante para que seja possível treinar modelos e avaliar os resultados em conjuntos externos ao treinamento, permitindo uma análise

```
# Construindo pipeline
audio_fe_pipeline = Pipeline([
    ('ae', AmplitudeEnvelop(frame_size=FRAME_SIZE, hop_length=HOP_LENGTH,
                            feature_aggreg=FEATURE_AGGREG)),
    ('rms', RMSEnergy(frame_size=FRAME_SIZE, hop_length=HOP_LENGTH,
                      feature_aggreg=FEATURE_AGGREG)),
    ('zcr', ZeroCrossingRate(frame_size=FRAME_SIZE, hop_length=HOP_LENGTH,
                             feature_aggreg=FEATURE_AGGREG)),
    ('ber', BandEnergyRatio(frame_size=STFT_FRAME_SIZE, hop_length=HOP_LENGTH, sr=SAMPLE_RATE,
                            split_freq=SPLIT_FREQ_BER, feature_aggreg=FEATURE_AGGREG)),
    ('sc', SpectralCentroid(frame_size=FRAME_SIZE, hop_length=HOP_LENGTH, sr=SAMPLE_RATE,
                            feature_aggreg=FEATURE_AGGREG)),
    ('bw', BandWidth(frame_size=FRAME_SIZE, hop_length=HOP_LENGTH, sr=SAMPLE_RATE,
                     feature_aggreg=FEATURE_AGGREG)),
    ('spec', GroupSpecAggreg(frame_size=FRAME_SIZE, hop_length=HOP_LENGTH, sr=SAMPLE_RATE,
                             split_freq=SPLIT_FREQ_BER, feature_aggreg=FEATURE_AGGREG)),
    ('mfcc', MFCCsAggreg(n_mfcc=N_MFCC, order=ORDER_MFCC, feature_aggreg=FEATURE_AGGREG))
])

# Gerando novamente base de dados
df = read_data(data_path=AUDIOS_PATH, sr=SAMPLE_RATE, signal_col=SIGNAL_COL, target_col=TARGET_COL)
X, y = data_pre_processing(df=df, signal_col=SIGNAL_COL, target_col=TARGET_COL)

# Aplicando transformação
X_prep = audio_fe_pipeline.fit_transform(X)
X_prep.drop('signal', axis=1, inplace=True)
print(f'Dimensões da base final resultante: {X_prep.shape}')
X_prep.head()
```

executed in 5m 47s, finished 16:26:38 2021-03-26

```
Dimensões da base final resultante: (528, 329)
```

| | ae_mean | ae_median | ae_std | ae_var | ae_max | ae_kurtosis | ae_skew | rms_engy_mean | rms_engy_median | rms_engy_std | rms_engy_var |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.026926 | 0.024551 | 0.025405 | 0.000645 | 0.100281 | -0.407265 | 0.657465 | 0.010282 | 0.006725 | 0.010507 | 0.000110 |
| **1** | 0.028720 | 0.021637 | 0.023985 | 0.000575 | 0.082886 | -0.722284 | 0.601592 | 0.009458 | 0.006850 | 0.008471 | 0.000072 |
| **2** | 0.028412 | 0.018982 | 0.029994 | 0.000900 | 0.117493 | 0.216276 | 1.022808 | 0.008077 | 0.004316 | 0.009053 | 0.000082 |
| **3** | 0.035016 | 0.030731 | 0.026640 | 0.000710 | 0.115021 | 0.366818 | 0.853834 | 0.011825 | 0.010346 | 0.009402 | 0.000088 |
| **4** | 0.046102 | 0.038208 | 0.043858 | 0.001923 | 0.208038 | 1.570043 | 1.269211 | 0.014332 | 0.010886 | 0.014519 | 0.000211 |

Figura 44 – Construção e aplicação do *pipeline* consolidado de preparação da base de dados contendo os elementos transformadores definidos previamente. Após a execução a base final gerada contém os agregados estatísticos para cada uma das características de sinais de áudio definidas.

real sobre o comportamento do sistema criado em situações práticas mais próximas de sua utilização cotidiana. Para isso, definiu-se uma proporcionalidade de 75% dos dados separados para o conjunto de treino, 15% para validação e 10% para teste. A Figura 45 abaixo traz a aplicação do método *train_test_split()* da biblioteca *sklearn* no DataFrame *X_prep* gerado logo após a aplicação do *pipeline* de preparação detalhado previamente.

Em seguida, definiu-se então uma lista de algoritmos de aprendizado de máquina a serem utilizados no treinamento e na avaliação do modelo de reconhecimento de locutores a partir de suas vozes. Com isso, uma maior gama de possibilidades se faz presente, possibilitando assim a avaliar e a seleção do melhor algoritmo capaz de realizar a tarefa determinada com o melhor desempenho. Em se tratando de uma modelagem multiclasse, ou seja, a variável $y$ da base de dados contém múltiplos elementos (locutores) a serem identificados, os modelos selecionados previamente para as etapas de treinamento são detalhados na Tabela 9.

```
# Gerando bases de treino, validação e teste
X_train, X_test, y_train, y_test = train_test_split(X_prep, y, test_size=1-TRAIN_RATIO,
                                                    random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_test, y_test, test_size=TEST_RATIO/(TEST_RATIO+VAL_RATIO),
                                                random_state=42)

# Verificando dimensões
print(f'Dimensões de X_train: {X_train.shape} - {round(100 * (X_train.shape[0] / len(df)), 2)}% do total')
print(f'Dimensões de X_val: {X_val.shape} - {round(100 * (X_val.shape[0] / len(df)), 2)}% do total')
print(f'Dimensões de X_test: {X_test.shape} - {round(100 * (X_test.shape[0] / len(df)), 2)}% do total')
```

executed in 8ms, finished 09:31:01 2021-03-27

```
Dimensões de X_train: (396, 329) - 75.0% do total
Dimensões de X_val: (79, 329) - 14.96% do total
Dimensões de X_test: (53, 329) - 10.04% do total
```

Figura 45 – Separação do conjunto de dados *X_prep* já preparado em três diferentes subconjuntos: treino, validação e teste. As proporções reais de representatividade volumétrica de cada subconjunto pode ser visualizada logo após a separação.

Como ponto de partida, decidiu-se treinar os algoritmos em suas respectivas configurações padrão em relação a hiperparâmetros. Apesar de esta estratégia contribuir com uma possível diminuição no poder de generalização dos modelos, isto é, um aumento do *overfitting* (efeito de sobreajuste gerado quando o algoritmo treinado apresenta um alto desempenho apenas nos dados aos quais o treinamento foi submetido), a escolha por esse formato tem como motivação o rápido treinamento e a avaliação de algoritmos candidatos, permitindo assim a posterior escolha e otimização de hiperparâmetros de um único algoritmo com desempenho satisfatório.

Tabela 9 – Algoritmos de aprendizado de máquina utilizados na tarefa de classificação multiclasse para identificação de locutores a partir das vozes.

| Algoritmo | Descrição |
|---|---|
| DecisionTreeClassifier() | Algoritmo de árvores de decisão considerado como um dos modelos mais intuitivos e de fácil interpretação, sendo possível visualizar todas as regras aplicadas pelo algoritmo até a tomada final da decisão (27). |
| RandomForestClassifier() | Florestas aleatórias compostas por diversas árvores de decisão. Trata-se de um dos algoritmos mais poderos de aprendizado de máquina, porém seu custo é maior e sua interpretabilidade menor em relação às árvores de decisão (27). |
| LGBMClassifier() | O algoritmo Light GBM é um *framework* de *gradient boosting* baseado em modelos de árvore. Trata-se de um algoritmo rápido, eficiente e utilizado em larga escala na comunidade (41). |

Definidos os algoritmos a serem utilizados na tarefa, a Figura 46 traz o código necessário para preparação dos objetos necessários para alimentar a classe *ClassificadorMulticlass* da biblioteca *pycomp*. A partir dela, é possível notar a importação das classes relacionadas aos próprios modelos preditivos, além da construção do dicionário Python representado pela variável *set_classifiers*, sendo esta responsável por armazenar os objetos atrelados a cada um dos modelos e, opcionalmente, hiperparâmetros que podem

ser utilizados na otimização. Como informado anteriormente, neste primeiro momento, as instâncias dos algoritmos serão utilizadas em seu formato mais puro e o treinamento ocorrerá sem a busca no espaço multidimensional de hiperparâmetros.

```
# Importando modelos
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from lightgbm import LGBMClassifier

# Instanciando objetos
dtree = DecisionTreeClassifier()
forest = RandomForestClassifier()
lgbm = LGBMClassifier(objective='multiclass', num_class=N_CLASSES)

# Criando dicionário set_classifiers
model_obj = [dtree, forest, lgbm]
model_names = [type(model).__name__ for model in model_obj]
set_classifiers = {name: {'model': obj, 'params': {}} for (name, obj) in zip(model_names, model_obj)}

print(f'Classificadores a serem treinados e avaliados: \n\n{model_names}')
```

executed in 9ms, finished 11:22:10 2021-03-27

```
Classificadores a serem treinados e avaliados:

['DecisionTreeClassifier', 'RandomForestClassifier', 'LGBMClassifier']
```

Figura 46 – Preparação dos algoritmos e das estruturas necessárias para alimentar a classe responsável por realizar todo o trabalho de treinamento e avaliação dos modelos de reconhecimento de locutores.

Neste cenário, é correto afirmar que a inclusão de novos algoritmos nesta etapa de treinamento e avaliação pode simplesmente ser realizada a partir da adição dos elementos presentes no algoritmo como, por exemplo, a importação da biblioteca e a instância do objeto atrelado à biblioteca, além da adição deste novo elemento na lista *model_obj* responsável por alocar todas as instâncias de algoritmos a serem utilizadas na construção do dicionário *set_classifiers*.

Uma vez preparada a estrutura de modelagem a partir de objetos específicos, é possível agora usufruir das funcionalidades da biblioteca *pycomp* a partir de sua classe *ClassificadorMulticlasse*, construída para suprir as necessidades exigidas por este próprio trabalho de graduação. A partir da Figura 47 é possível observar a criação do objeto *trainer* e a execução de seu método *fit()*, sendo este responsável pelo treinamento dos algoritmos selecionados. Visando facilitar o processo de análise, mensagens de log são geradas de acordo com o andamento do processo.

```
# Instanciando objeto e treinando modelos
trainer = ClassificadorMulticlasse(encoded_target=ENCODED_TARGET)
trainer.fit(set_classifiers, X_train, y_train, random_search=False)
```

executed in 1.78s, finished 11:22:14 2021-03-27

```
DEBUG;2021-03-27 11:22:13;trainer.py;trainer;1606;Treinando modelo DecisionTreeClassifier
DEBUG;2021-03-27 11:22:13;trainer.py;trainer;1606;Treinando modelo RandomForestClassifier
DEBUG;2021-03-27 11:22:13;trainer.py;trainer;1606;Treinando modelo LGBMClassifier
```

Figura 47 – Criação do objeto *trainer* a partir da classe *ClassificadorMulticlasse* e realização do processo de treinamento a partir da execução do método *fit()*.

Com isso, tem-se então algoritmos de aprendizado já treinados dentro do propósito de reconhecimento de voz. Os objetos associados se encontram armazenados no objeto *trainer* e podem ser resgatados a qualquer momento a partir de métodos *getters*. Como foram fornecidas bases contendo características extraídas de sinais de áudio vinculadas a uma variável resposta indicativa do locutor, todo esse processo pode ser caracterizado como um aprendizado de máquina supervisionado em uma abordagem multiclasse. Como próximos passos, os resultados obtidos nesta etapa de treinamento serão avaliados em critérios comparativos entre os algoritmos testados.

# 4 Resultados

Após toda a etapa de preparação e treinamento de modelos de aprendizado de máquina com o intuito de reconhecer locutores a partir de suas vozes, é chegado o momento de avaliar o desempenho obtido de modo a validar a eficiência dos modelos dentro do cenário proposto. Nesta sessão, será possível observar os resultados através das principais métricas de avaliação de modelos de classificação, bem como matrizes de confusão e análises detalhadas sobre as características de áudio mais relevantes para cada modelo treinado.

## 4.1 Métricas de Classificação

Considerando a separação do conjunto de dados em subconjuntos de treino, validação e teste, as métricas de classificação computadas para cada um dos modelos treinados têm como base a utilização do subconjunto de treino a partir da aplicação de validação cruzada e do subconjunto de validação em uma avaliação direta. A biblioteca *sklearn* fornece uma implementação da validação cruzada através das funções *cross_val_score()* e *cross_val_predict()* onde, em cada uma delas, é preciso definir um parâmetro relacionado à quantidade de *K-folds* de separação do conjunto.

Assim, a Tabela 10 traz uma relação das principais métricas de avaliação de modelos de classificação para cada um dos algoritmos treinados e para cada uma das classes (locutores) presentes na base. Cada combinação é avaliada nas duas propostas acima exemplificadas: validação cruzada com dados de treino e avaliação direta com dados de validação. Neste contexto, a validação cruzada foi aplicada utilizando 5 *K-folds*.

Analisando a os dados disponibilizados na Tabela 10, é possível perceber que o modelo *LightGBM* apresentou desempenho superior se comparado aos modelos *DecisionTrees* e *RandomForest*. É interessante citar que, nos dados de validação, o modelo treinado com *LightGBM* apresentou 100% de desempenho em todas as métricas de classificação avaliadas para todos as diferentes classes envolvidas.

De fato, o resultado obtido para os modelos candidatos foi extremamente elevado e altamente satisfatório. Considerando o contexto de gravação e obtenção dos áudios utilizados no treinamento, é possível afirmar que os modelos treinados, mesmo que em suas configurações padrão, funcionam excepcionalmente bem dentro do propósito de identificação de locutores a partir de vozes no cenário restrito por este trabalho de graduação.

Uma outra forma imensamente eficiente de avaliar a resposta de modelos de classificação é a partir da matriz de confusão. A partir dessa ferramenta, pode-se analisar

Tabela 10 – Performance dos modelos de aprendizado de máquina treinados.

| Modelo | Abordagem | Classe | Acurácia | Precisão | Recall | F1 |
|---|---|---|---|---|---|---|
| DecisionTree | Treino 5 K-folds | Locutor 01 | 0.9419 | 0.8913 | 0.8723 | 0.8817 |
| DecisionTree | Treino 5 K-folds | Locutor 02 | 0.9419 | 0.9484 | 0.9200 | 0.9340 |
| DecisionTree | Treino 5 K-folds | Locutor 03 | 0.9419 | 0.9313 | 0.9500 | 0.9405 |
| DecisionTree | Treino 5 K-folds | Locutor 04 | 0.9419 | 0.9602 | 0.9731 | 0.9666 |
| DecisionTree | Validação | Locutor 01 | 0.9620 | 1.0000 | 1.0000 | 1.0000 |
| DecisionTree | Validação | Locutor 02 | 0.9620 | 0.8421 | 1.0000 | 0.9142 |
| DecisionTree | Validação | Locutor 03 | 0.9620 | 1.0000 | 0.8636 | 0.9268 |
| DecisionTree | Validação | Locutor 04 | 0.9620 | 1.0000 | 1.0000 | 1.0000 |
| RandomForest | Treino 5 K-folds | Locutor 01 | 0.9898 | 0.9591 | 1.0000 | 0.9791 |
| RandomForest | Treino 5 K-folds | Locutor 02 | 0.9898 | 1.0000 | 0.9800 | 0.9898 |
| RandomForest | Treino 5 K-folds | Locutor 03 | 0.9898 | 1.0000 | 0.9800 | 0.9898 |
| RandomForest | Treino 5 K-folds | Locutor 04 | 0.9898 | 0.9867 | 1.0000 | 0.9933 |
| RandomForest | Validação | Locutor 01 | 0.9873 | 1.0000 | 1.0000 | 1.0000 |
| RandomForest | Validação | Locutor 02 | 0.9873 | 1.0000 | 0.9375 | 0.9677 |
| RandomForest | Validação | Locutor 03 | 0.9873 | 1.0000 | 1.0000 | 1.0000 |
| RandomForest | Validação | Locutor 04 | 0.9873 | 0.9677 | 1.0000 | 0.9836 |
| LightGBM | Treino 5 K-folds | Locutor 01 | 0.9924 | **1.0000** | 0.9787 | 0.9892 |
| LightGBM | Treino 5 K-folds | Locutor 02 | 0.9924 | **1.0000** | 0.9800 | 0.9898 |
| LightGBM | Treino 5 K-folds | Locutor 03 | 0.9924 | 0.9803 | **1.0000** | 0.9900 |
| LightGBM | Treino 5 K-folds | Locutor 04 | 0.9924 | 0.9933 | **1.0000** | 0.9966 |
| LightGBM | Validação | Locutor 01 | **1.0000** | **1.0000** | **1.0000** | **1.0000** |
| LightGBM | Validação | Locutor 02 | **1.0000** | **1.0000** | **1.0000** | **1.0000** |
| LightGBM | Validação | Locutor 03 | **1.0000** | **1.0000** | **1.0000** | **1.0000** |
| LightGBM | Validação | Locutor 04 | **1.0000** | **1.0000** | **1.0000** | **1.0000** |

os erros e acertos associados a cada uma das classes de um modelo de classificação treinado, sendo possível verificar erros sistêmicos de viés que podem estar associados a uma ou mais classes específicas. A Figura 48 traz matrizes de confusão construídas sob métodos da biblioteca *pycomp* em conjunto com a biblioteca *sklearn*. Nela, são propostas duas matrizes de confusão para cada um dos modelos treinados sendo, à esquerda, posicionada matrizes geradas a partir da base de treino utilizando validação cruzada e, à direita, matrizes geradas utilizando diretamente a base de validação.

No eixo $y$ de cada matriz, tem-se a resposta oficial do modelo associado a cada entrada na base de dados (treino ou validação). Já no eixo $x$, é possível encontrar as respostas retornadas pelos modelos para cada uma das classes. Dessa forma, é correto dizer que a diagonal descendente da matriz representa os acertos obtidos em cada uma das classes, visto que a classe presente no eixo horizontal cruzada com a classe presente no eixo vertical indica que o modelo acertou a resposta correta atrelada ao registro. Elementos fora dessa diagonal descendente indicam erros de resposta do modelo e, a partir deles, é possível observar, em cada caso, qual a classe verdadeiramente esperada dentro da classe

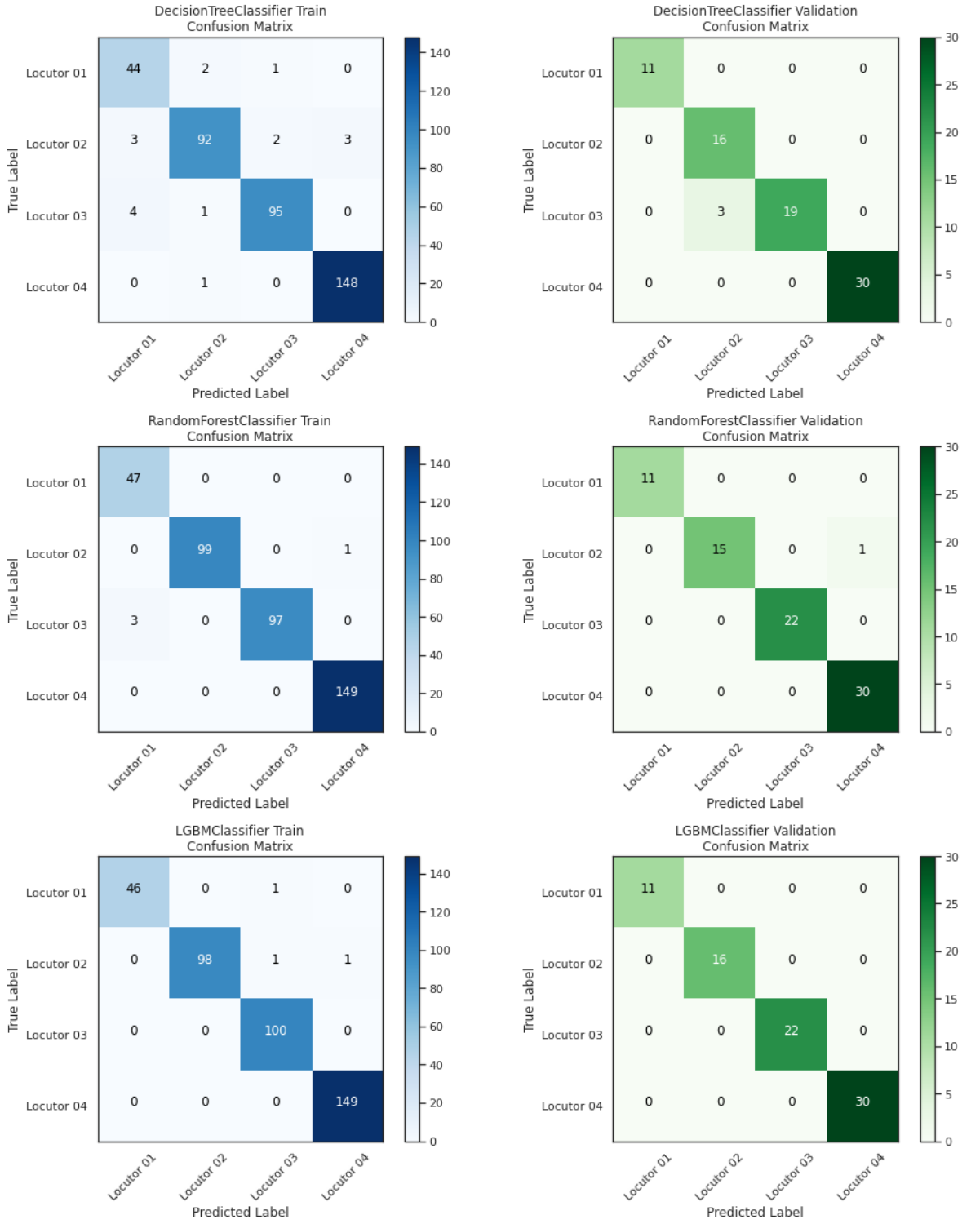


Figura 48 – Matrizes de confusão construídas para cada um dos algoritmos de aprendizado treinados.

de resposta retornada pelos modelos.

Ainda sobre a Figura 48, a análise da matriz de confusão do modelo *LightGBM* permite verificar a presença de poucos erros atrelados à base de treino e nenhum erro obtido utilizando a base de validação.

## 4.2 Características de Áudio mais Relevantes

Uma das etapas cruciais elegíveis neste trabalho de graduação é a verificação das características mais relevantes dentro do contexto de identificação de locutores a partir de sinais de áudio fornecidos como entrada. Nos passos anteriores, foi possível extrair e consolidar uma base de modelagem com 329 características distintas (maiores detalhes podem ser encontrados na Tabela 7). Em um caráter totalmente explorativo, a seleção das características e agregados estatísticos considerados dentro da fase de preparação da base atuaram como uma forma de possibilitar, após o treinamento, a verdadeira análise em torno do que realmente é relevante e do que pode possivelmente ser descartado dentro do objetivo de otimização dos algoritmos treinados.

Tecnicamente, grande parte dos algoritmos de classificação trazem consigo métodos intrínsecos capazes de retornar indicadores de importância de cada uma das variáveis recebidas por um modelo já treinado. Neste cenário, uma variável ou *feature* pode ser considerada relevante quando esta tem impacto direto na resposta do modelo, permitindo, desse modo, que a identificação de cada uma das classes presentes no problema de otimização seja feita de forma mais eficiente. Assim, considerando o modelo *LightGBM* treinado na etapa anterior, a Figura 49 traz uma relação das 40 principais características de áudio mais importantes consideradas por este algoritmo.

Observando a Figura 49, é interessante notar a grande representatividade de características de sinais de áudio relacionadas aos MFCCs. Das primeiras 40 carcaterísticas mais relevantes obtidas no cenário exemplificado, 29 possuem relação com os MFCCs, seja na versão mais pura ou em suas formas obtidas pela primeira e segunda derivadas. A isso pode ser associado o fato de que tais componentes são essencialmente obtidas a partir da simulação dos fonemas fundamentais da geração de fala humana, sendo então uma ferramenta extremante importante na caracterização de diferentes vozes.

Algumas outras características testadas que podem ser destacadas dentro da relevância de identificação de locutores encontra-se o máximo do centroide espectral (representada pela nomenclatura *sc_max*), a potência média de altas frequências (indicada por *high_freq_pwr_mean*) e a média do envelope de amplitude (dada por *ae_mean*). Este tipo de análise permite pontuar que, nos termos restringidos por este trabalho de graduação, existem outras características relevantes, além das MFCCs, que contribuem positivamente para a devida identificação de diferentes locutores utilizando suas respectivas vozes como insumos.

Analisar a importância das *features* em um contexto de relevância para a resposta de algoritmos já treinados é substancialmente importante para otimizar processamento ou até mesmo o desempenho de modelos. Entretanto, neste tipo de análise, ainda não é possível concluir impactos atrelados à magnitude de cada uma das características observadas. Em

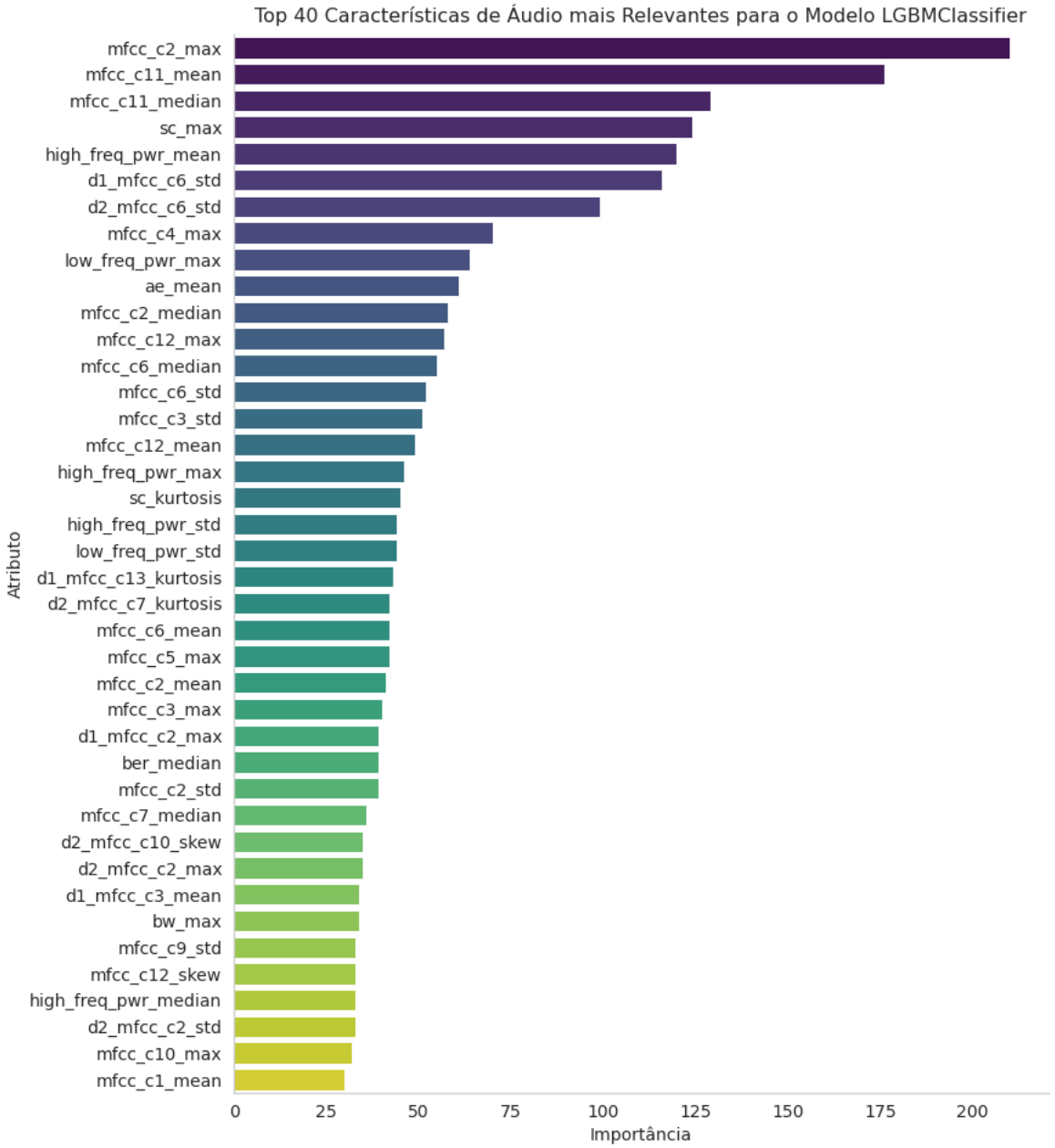


Figura 49 – Principais características extraídas de sinais de áudio em termos de relevância frente ao objetivo de identificação de locutores para o modelo *LightGBM* já treinado.

outras palavras, foi possível verificar, a partir da Figura 49, a grande presença de variáveis relacionadas aos MFCCs porém, de forma direta, não é possível deduzir os efeitos deste impacto em termos da magnitude de cada um desses coeficientes. Para isso, uma das ferramentas utilizada em maior escala no universo de modelagem e interpretabilidade de modelos preditivos é a análise *shap*. De maneira resumida, é possível dizer que a análise *shap* permite avaliar como cada valor de cada variável influenciou no resultado alcançado pelo modelo preditivo (42).

Por fim, analisadas as principais métricas e o resultado obtido por cada um dos

algoritmos de aprendizado de máquina elegíveis para o treinamento, tanto na base de treino, quanto na base de validação, é possível utilizar a base de teste reservada para uma verificação final em um modelo único selecionado. Considerando o desempenho obtido em cada caso, o modelo *LightGBM* será escolhido como algoritmo final devido à sua eficiência e à sua rapidez na resposta. Assim, a Tabela 11 traz os resultados do modelo acima pontuado na base final de testes.

Tabela 11 – Desempenho final do modelo *LightGBM* na base de testes reservada para esta finalidade.

| Modelo | Abordagem | Classe | Acurácia | Precisão | Recall | F1 |
|---|---|---|---|---|---|---|
| LightGBM | Teste | Locutor 01 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| LightGBM | Teste | Locutor 02 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| LightGBM | Teste | Locutor 03 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| LightGBM | Teste | Locutor 04 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |

Observando a Tabela 11, nota-se um acerto de 100% para todos os locutores presentes na base de modelagem e em todas as métricas medidas. Alguns dos principais fatores que podem ter influenciado positivamente no alcance desses resultados:

- Uniformidade do processo de gravação dos sinais de áudio, considerando:
  - Utilização de um mesmo hardware para todas as gravações
  - Gravações realizadas em um ambiente silencioso e com controle de ruído externo
  - Definição de processo de qualidade para manutenção apenas de áudios limpos e com pronúncia fluida dos locutores
- Utilização de uma quantidade relativamente grande de áudios no processo de modelagem (visualizar Figuras 29 e 30 da seção 3.2)
- Utilização de áudios validados de uma UBM confiável (*Mozilla Common Voice*)
- Treinamento de modelo com uma grande quantidade de atributos extraídos (329 características ao todo)

# 5 Conclusão

Após a aplicação de todas as etapas presentes na Metodologia deste trabalho de graduação, é chegado o momento de avaliar a conclusão dos objetivos definidos e o alcance dos resultados almejados.

No contexto de utilização de uma abordagem clássica de aprendizado de máquina para a construção de um modelo capaz de reconhecer locutores a partir de suas respectivas vozes, fez-se necessário, como ponto de partida, um entendimento claro sobre as características de sinais de áudio a serem extraídas para essa finalidade. Em um cenário alternativo, a aplicação de aprendizado profundo (*Deep Learning*) a partir de redes neurais poderia servir como uma forma interessante de construir um modelo extremamente eficiente de reconhecimento de locutores. Porém, em geral, sob este ponto de vista os próprios sinais brutos de áudio seriam utilizados como entrada, deixando a cargo das camadas da rede neural a extração de relações que, em muitos casos, poderiam ser pouco interpretáveis. De fato, um dos principais objetivos deste trabalho de graduação teve como base a navegação e a exploração das características de um sinal de fala comum, permitindo a criação de um vasto leque de possibilidades a partir deste entendimento. Uma das vertentes presentes neste leque pôde ser exemplificada com a análise das principais variáveis que permitem uma eficiente identificação de locutores.

Dito isso, foi possível construir um *pipeline* completo e automático de extração de características de áudio definidas previamente de acordo com direcionamentos encontrados na vasta literatura consultada. A partir da linguagem Python, seus *frameworks* e suas bibliotecas (com destaque à LibROSA), criou-se uma verdadeira documentação das etapas associadas à obtenção de cada uma das características elegíveis, permitindo análises técnicas e visuais dos elementos obtidos dentro de sinais de áudio originalmente disponibilizados no domínio do tempo. Este processo foi essencial para proporcionar um pleno entendimento sobre o significado de cada uma das variáveis vinculadas ao processo de extração, sejam estas presentes no domínio do tempo, no domínio da frequência ou mesmo no domínio tempo-frequência.

Assim, considerando os resultados obtidos na seção equivalente, foi possível inserir a base estruturada de dados gerada pelo *pipeline* de extração em algoritmos de aprendizado de máquina com uma abordagem de classificação multiclasse. Dessa forma, o desempenho evidenciado pela Tabela 10 permite afirmar que o reconhecimento de locutores pôde ser obtido com um alto grau de confiabilidade. Para comprovar essa afirmação, a Tabela 11 mostra que o modelo *LightGBM* selecionado acertou em 100% dos casos disponíveis em uma base de teste utilizada justamente com o propósito de validação final, o que, de fato,

é um resultado altamente satisfatório.

Em complemento à eficiência apresentada pela etapa de modelagem, a Figura 49 permite avaliar as principais características em termos de impacto no resultado do modelo e, considerando as primeiras 40 variáveis mais relevantes, é possível observar uma grande representividade dos coeficientes MFCCs como forma de identificação de locutores. Além disso, uma análise mais detalhada permite concluir que outras características de áudio, em alguns agregados específicos, contribuem positivamente para essa identificação. São exemplos: o envelope de amplitude, o centroide espectral e o espectrograma de potências.

Por fim, é preciso reforçar o contexto de geração dos dados como um processo estabelecido por gravações realizadas em circunstâncias restringidas pelo *hardware* utilizado na captura e limiar de ruídos externos presentes no ambiente. Todos os áudios foram gravados com o mesmo dispositivo móvel em um ambiente noturno, considerado silencioso. A uniformidade adicionada à construção da base de dados reflete diretamente na aplicação do modelo treinado em um cenário real de utilização, e fatores que diferem dos restringidos durante a gravação podem impactar significamente na eficiência do modelo.

No mais, analisando todas as ferramentas propostas e os elementos gerados por este trabalho de graduação, é possível concluir que o resultado almejado foi alcançado com sucesso. A exploração dos sinais de áudio e a construção de uma metodologia própria de obtenção dos insumos necessários para a uniformidade do processo foram fatores cruciais para a construção de modelos preditivos capazes de utilizar um fator biologicamente humano, como a voz, em cenários tecnológicos de reconhecimento.

Com isso, o autor encerra este trabalho e espera, de forma sincera, ter contribuído significantemente com a comunidade de tecnologia e, principalmente, com quem futuramente se interessar pelo tema aqui abordado. Detalhes adicionais sobre o projeto, bem como fluxos e scripts associados ao consumo da inteligência aqui desenvolvida podem ser encontrados na página do projeto no Github (43).

# Referências

1 LATINUS, M.; BELIN, P. Human voice perception. *Current Biology*, v. 21, n. 4, p. R143–R145, 2011. ISSN 0960-9822. Disponível em: <https://www.sciencedirect.com/science/article/pii/S096098221001701X>.

2 BELIN, P.; FECTEAU, S.; BéDARD, C. Thinking the voice: neural correlates of voice perception. *Trends in Cognitive Sciences*, v. 8, n. 3, p. 129–135, 2004. ISSN 1364-6613. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1364661304000257>.

3 SIANTIKOS, G.; GIANNAKOPOULOS, T.; KONSTANTOPOULOS, S. Monitoring activities of daily living using audio analysis and a raspberrypi: A use case on bathroom activity monitoring. In: *Information and Communication Technologies for Ageing Well and e-Health*. [S.l.]: Spring International Publishing, 2017. p. 20–35.

4 VUEGEN, L. et al. Automatic monitoring of activities of daily living based on real-life acoustic sensor data: a preliminary study. *KU Leuven*, Association for Computational Linguistics (ACL); Stroudsburg, p. 113–118, 2013. Disponível em: <https://lirias.kuleuven.be/1671176?limo=0>.

5 HRUSHIKESH, M. et al. Implementation of mood detection through voice analysis using librosa and cnn. *IRJET*, Association for Computational Linguistics (ACL); Stroudsburg, v. 7, p. 5876–5879, 2020. ISSN 2395-0056. Disponível em: <https://www.irjet.net/archives/V7/i6/IRJET-V7I61106.pdf>.

6 SHADIEV, R.; HUANG, Y.-M. Facilitating cross-cultural understanding with learning activities supported by speech-to-text recognition and computer-aided translation. *Computers  Education*, v. 98, p. 130–141, 2016. ISSN 0360-1315. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0360131516300768>.

7 HANSEN, J. H. L.; HASAN, T. Speaker recognition by machines and humans: A tutorial review. *IEEE Signal Processing Magazine*, v. 32, n. 6, p. 74–99, 2015.

8 CAMPBELL, W. M. et al. Svm based speaker verification using a gmm supervector kernel and nap variability compensation. In: *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*. [S.l.: s.n.], 2006. v. 1, p. I–I.

9 NICKOLLS, J.; DALLY, W. J. The gpu computing era. *IEEE Micro*, v. 30, n. 2, p. 56–69, 2010.

10 LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, p. 436–444, 2015.

11 MCFEE, B. et al. librosa: Audio and music signal analysis in python. *Proc. of the 14th Python in science Conf*, 2015.

12 KëPUSKA, V.; BOHOUTA, G. Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home). In: *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. [S.l.: s.n.], 2018. p. 99–103.

13 REYNOLDS, D. A.; QUATIERI, T. F.; DUNN, R. B. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, v. 10, n. 1, p. 19–41, 2000. ISSN 1051-2004. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1051200499903615>.

14 DODDINGTON, G. R. *Speaker recognition—Identifying people by their voices*. 1985. 1651-1664 p.

15 KINNUNEN, T.; LI, H. An overview of text-independent speaker recognition: From features to supervectors. *Speech Communication*, v. 52, n. 1, p. 12–40, 2010. ISSN 0167-6393. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167639309001289>.

16 MOZILLA. *Common Voice*. 2021. Disponível em: <https://commonvoice.mozilla.org/pt>.

17 PELGROM, M. *Analog-to-Digital Conversion*. [S.l.]: Springer, 2017. v. 1. ISBN 978-3-319-44970-8.

18 RUSS, A. L. *An introduction to audio content analysis: applications in signal processing and music informatics*. [S.l.]: IEEE Press, 2012. v. 1. ISBN 978-1-118-26682-3.

19 RUSS, M. *Sound Synthesis and Sampling*. [S.l.]: Elsevier, 2009. v. 1. ISBN 978-0-240-52105-3.

20 Audacity Team. *Sample Rates*. 2021. Disponível em: <https://alphamanual.audacityteam.org/man/Sample_Rates>.

21 KNEES, P.; SCHEDL, M. *Music similarity and retrieval: an introduction to audio and web-based strategies*. [S.l.]: Springer, 2016. v. 1. ISSN 1387-5264. ISBN 978-3-662-49722-7.

22 LOKESH, S.; DEVI, M. R. Analog-to-digital converter survey and analysis. *Cluster Comput*, v. 22, p. 11669–11679, 2017.

23 FAYEK, H. *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficientes (MFCCs) and What's In-Between*. 2016. Disponível em: <https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html>.

24 HANIFA, R. M.; ISA, K.; MOHAMAD, S. A review on speaker recognition: Technology and challenges. *Computers  Electrical Engineering*, v. 90, p. 107005, 2021. ISSN 0045-7906. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0045790621000318>.

25 OPPENHEIM, A. V.; SCHAFER, R. W. From frequency to quefrency: a history of the cepstrum. *IEEE Signal Processing Magazine*, v. 21, n. 5, p. 95–106, 2004.

26 MCFEE, B. et al. *librosa: 0.8.0*. 2020. Disponível em: <https://librosa.org/doc/main/index.html>.

27 GERON, A. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. [S.l.]: O'Reilly, 2017. v. 1. ISBN 978149196229.

28 NG, A. *Machine Learning*. 2019. Disponível em: <https://www.coursera.org/learn/machine-learning>.

29 PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

30 BUITINCK, L. et al. API design for machine learning software: experiences from the scikit-learn project. In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. [S.l.: s.n.], 2013. p. 108–122.

31 BOISBERRANGER, J. du et al. *Scikit-learn supervised learning*. 2020. Disponível em: <https://scikit-learn.org/stable/supervised_learning.html#supervised-learning>.

32 GUANGA, A. *Understand classification performance metrics*. Becoming Human: Artificial Intelligente Magazine, 2018. Disponível em: <https://becominghuman.ai/understand-classification-performance-metrics-cad56f2da3aa>.

33 SUNASRA, M. *Performance metrics for classification problems in machine learning*. Medium, 2018. Disponível em: <https://medium.com/@MohammedS/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b>.

34 GEORGE, O. *Common classification model evaluation metrics*. Towards Data Science, 2019. Disponível em: <https://towardsdatascience.com/common-classification-model-evaluation-metrics-2ba0a7a7436e>.

35 BOISBERRANGER, J. du et al. *Metrics and scoring: quantifying the quality of predictions*. 2020. Disponível em: <https://scikit-learn.org/stable/modules/model_evaluation.html>.

36 Quality Apps. *Voice Recorder*. Disponível em: <https://play.google.com/store/apps/details?id=com.media.bestrecorder.audiorecorder&hl=pt_BR&gl=BR>.

37 Abc Apps. *Decibelímetro (Sound Meter)*. Disponível em: <https://play.google.com/store/apps/details?id=com.gamebasic.decibel&hl=pt_BR&gl=BR>.

38 The pandas development team. *pandas-dev/pandas: Pandas*. Zenodo, 2020. Disponível em: <https://doi.org/10.5281/zenodo.3509134>.

39 HARRIS, C. R. et al. Array programming with NumPy. *Nature*, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, set. 2020. Disponível em: <https://doi.org/10.1038/s41586-020-2649-2>.

40 PANINI, T. *Pycomp*. Disponível em: <https://github.com/ThiagoPanini/pycomp>.

41 KE, G. et al. Lightgbm: A highly efficient gradient boosting decision tree. In: GUYON, I. et al. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. v. 30. Disponível em: <https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf>.

42 LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. In: GUYON, I. et al. (Ed.). *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017. p. 4765–4774. Disponível em: <http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf>.

43 PANINI, T. *Github Voice Unlocker*. Disponível em: <https://github.com/ThiagoPanini/voice-unlocker>.