

Universidade Estadual Paulista  
Instituto de Biociências, Letras e Ciências Exatas  
Departamento de Ciência da Computação e Estatística

Mateus Marques Montagnoli

Detecção de ataques de replays a sistemas de  
verificação automática de locutores (ASV).

São José do Rio Preto - SP

2021

Mateus Marques Montagnoli

Detecção de ataques de replays a sistemas de  
verificação automática de locutores (ASV).

Monografia apresentada ao Programa de  
graduação em Ciência da Computação da  
UNESP para obtenção do título de Bacharel.

Orientador: Aleardo Manacero Junior

São José do Rio Preto - SP

2021

AUTORIZO A DIVULGAÇÃO TOTAL OU PARCIAL DESTE TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTES

M758d Montagnoli, Mateus Marques  
Detecção de ataques de replays a sistemas de verificação automática de locutores (ASV). / Mateus Marques Montagnoli. -- São José do Rio Preto, 2021  
48 p. : il., tabs.

Trabalho de conclusão de curso (Bacharelado - Ciência da Computação) -  
Universidade Estadual Paulista (Unesp), Instituto de Biociências Letras e  
Ciências Exatas, São José do Rio Preto  
Orientador: Aleardo Manacero Junior

1. Processamento de sinais. 2. Reconhecimento de locutor. 3. Biometria. 4.  
Ataque via gravação e reprodução de voz. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Mateus Marques Montagnoli

Detecção de ataques de replays a sistemas de  
verificação automática de locutores (ASV).

Monografia apresentada ao Programa de  
graduação em Ciência da Computação da  
UNESP para obtenção do título de Bacharel.

Comissão Examinadora

Prof. Dr. Aleardo Manacero Junior  
UNESP – Câmpus de São José do Rio Preto  
Orientador

Prof. Dr. Carlos Roberto Valêncio  
UNESP – Câmpus de São José do Rio Preto

Prof. Dr. Geraldo Francisco Donega Zafalon  
UNESP – Câmpus de São José do Rio Preto

São José do Rio Preto - SP  
10 de Janeiro de 2021

# Resumo

Neste trabalho, são propostos dois modelos capazes de detectar ataques gravação e reprodução de voz em sistemas de verificação automática de locutores (ASV). Os modelos consistem em um classificador de k-vizinho mais próximo (k-NN) e uma máquina de vetores de suporte (SVM), os quais serão treinados e validados a partir da extração das características dos sinais de voz presentes no banco de dados ASVspoof 2019, as características utilizadas são a Estimativa do Espectro de Potência e a *Teager Energy Operator*. Além disso, foi feita a análise de todas as qualidades de dispositivos de reprodução, onde o classificador k-NN obteve bons resultados trabalhando com qualidades baixas com 87,76% de acurácia média, enquanto que, o SVM se mostrou mais eficiente, com 97,21%, 87,37% e 71,23% de acurácia média em qualidades baixas, altas e todas as qualidades respectivamente.

Palavras-chave: Processamento de sinais. Reconhecimento de locutor. Biometria. Ataque via gravação e reprodução de voz.

# Abstract

In this work, are proposed two models that are able to detect replay spoofing attacks on Automatic Speaker Verification (ASV). The models consists on a k-Nearest Neighbour (k-NN) classifier and Support Vector Machine (SVM) which will be trained and evaluated from the extraction of the voice signals features from the ASVspoof 2019 data base, the features used are the Power Spectrum Estimation and Teager Energy Operator. Futhermore, the analysis of all varieties of the quality of reproduction devices was done, where the k-NN classifier got good result working in low qualities with 87.76% of average accuracy, meanwhile, SVM was more efficient showing 97.21%, 87.37% and 71.23% of average accuracy in low, high and all qualities respectively.

Keywords: Signal processing. Speaker recognition. Biometrics. Replay attack.

# Lista de Figuras

Figura 2.1 - Combinações possíveis de um CM e ASV. Em (i) e (ii) temos a apresentação do modelo cascata, onde os sistemas estão empregados um atrás do outro, CM e em sequência o ASV na primeira divisão, enquanto que, ASV e em sequência o CM na segunda divisão. Por último, temos que em (iii) a adoção do modelo paralelo. . . . .	18
Figura 2.2 - Visualização da DFT. . . . .	21
Figura 2.3 - Exemplo de uma classifica de k-NN com $k = 3$ , $k = 4$ e $k = 5$ . . . . .	29
Figura 2.4 - Exemplo de divisão de classes via Hiperplano. . . . .	31
Figura 2.5 - Aplicação de <i>kernel</i> em um conjunto de características linearmente inseparáveis. . . . .	32
Figura 3.1 - O fluxo do desenvolvimento do trabalho. . . . .	35
Figura 3.2 - Aplicação do filtro de Pré-Ênfase. . . . .	37
Figura 3.3 - Estimativa da densidade do espectro utilizando uma janela de 512 pontos. . . . .	39
Figura 3.4 - <i>Teager Energy Operator</i> de um sinal de entrada de 2048 pontos. . . . .	40

# Lista de Tabelas

Tabela 3.1 - Tabela de identificação de características do sinal. . . . .	36
Tabela 3.2 - Tabela de identificação de características do ataque. . . . .	36
Tabela 4.1 - Resultados obtidos pelo classificador k-NN, com $k = 10$ , para cada tipo de conjunto de treino e teste. QP para qualidade de <i>replay</i> perfeita, QA para qualidade de <i>replay</i> alta, QB para qualidade de <i>replay</i> baixa e Todas foi utilizados um conjunto de exemplos com as três qualidades. . . . .	43
Tabela 4.2 - Resultados obtidos pelo classificador SVM. para cada tipo de conjunto de treino e teste. QP para qualidade de <i>replay</i> perfeita, QA para qualidade de <i>replay</i> alta, QB para qualidade de <i>replay</i> baixa e Todas foi utilizados um conjunto de exemplos com as três qualidades. . . . .	44

# Lista de Abreviaturas

<b>ACC</b>	<i>Accuracy</i>
<b>ASV</b>	<i>Automatic Speaker Verification</i>
<b>CM</b>	<i>Spoofing Countermeasure</i>
<b>DFT</b>	<i>Discrete Fourier Transform</i>
<b>DWT</b>	<i>Discrete Wavelet Transform</i>
<b>EER</b>	<i>Equal Error Rate</i>
<b>IEEE</b>	<i>Institute of Electrical and Eletronic Engineers</i>
<b>K-NN</b>	<i>K Nearest Neighbour</i>
<b>PCM</b>	<i>Pulse Code Modulation</i>
<b>RIF</b>	<i>Resource Interchange File Format</i>
<b>RFC</b>	<i>Request for Comments</i>
<b>SVM</b>	<i>Support Vector Machine</i>
<b>TEO</b>	<i>Teager Energy Operator</i>
<b>TTS</b>	<i>Text-to-speech</i>
<b>VC</b>	<i>Voice Conversion</i>
<b>WAVE</b>	<i>Waveform</i>

# Sumário

<b>1</b>	<b>Introdução</b>	<b>13</b>
1.1	Considerações iniciais . . . . .	13
1.2	Objetivo . . . . .	14
1.3	Motivação . . . . .	14
1.4	Organização do trabalho . . . . .	15
1.5	Metodologia . . . . .	15
<b>2</b>	<b>Revisão Bibliográfica</b>	<b>17</b>
2.1	Anti-falsificação de voz ou <i>Speech Spoofing Countermeasure</i> . . . . .	17
2.2	Formato Wave . . . . .	19
2.3	Filtro de Pré-Ênfase . . . . .	19
2.4	Análise Espectral . . . . .	20
2.4.1	Transformada discreta de fourier ou <i>Discrete Fourier Transform</i> (DFT)	20
2.5	Extração de Características . . . . .	22
2.5.1	Estimativa da densidade do Espectro . . . . .	22
2.5.2	<i>Teager Energy Operator</i> (TEO) . . . . .	23
2.6	Validação cruzada . . . . .	26
2.7	Classificação . . . . .	27
2.7.1	k-vizinhos mais próximos ou k-Nearest Neighbors (k-NN) . . . . .	28
2.7.2	Maquina de Vetores de Suporte ou Support Vector Machine (SVM) . . . . .	30
2.8	Trabalhos relacionados . . . . .	33
<b>3</b>	<b>Detalhamento do Trabalho Proposto</b>	<b>35</b>
3.1	Considerações iniciais . . . . .	35

3.2	Base de Dados . . . . .	35
3.3	Pré-processamento . . . . .	37
3.4	Extração de características . . . . .	37
3.5	Classificação . . . . .	40
<b>4</b>	<b>Testes e Resultados</b>	<b>42</b>
4.1	Pontos Importantes . . . . .	42
4.2	Testes k-NN . . . . .	42
4.3	Testes SVM . . . . .	43
<b>5</b>	<b>Conclusões</b>	<b>45</b>
	<b>Referências</b>	<b>45</b>

# Capítulo 1

## Introdução

### 1.1 Considerações iniciais

O mundo conectado trouxe consigo algumas facilidades, como o acesso remoto a diversas aplicações que apresentam informações pessoais de um indivíduo. Como por exemplo, os aplicativos que oferecem acesso as informações privadas de um indivíduo, para acessar tal informações, obviamente, é necessário um sistema de autenticação, o qual irá identificar se a pessoa que está tentando acessar esse conteúdo é, de fato, a pessoa detentora desses dados. Para isso, o uso do reconhecimento biométrico vem ganhando espaço como forma de autenticação [7].

Um das formas de autenticação que vem ganhando espaço no cenário da biometria são os sistemas verificação por voz, conhecidos como ASV (Automatic Speaker Verification). Esses sistemas, já estão sendo utilizado no mercado [8] devido a facilidade que o usuário tem de realizar a autenticação com uma boa medida de proteção.

## 1.2 Objetivo

Este trabalho, tem como objetivo o estudo de conceitos e implementação de um sistema que consiga realizar a classificação de um sinal de voz, classificando-o como um sinal gravado ou sinal genuíno, ou seja, verificar se está ocorrendo um ataque no ASV. Dois modelos foram utilizados e comparados, k-NN e SVM. Além disso, a análise da performance da classificação foi realizada.

## 1.3 Motivação

Assim como outros sistemas de autenticação, o ASV não está livre de possíveis ataques que tentam explorar vulnerabilidades presentes no sistema para tentar ganhar o acesso as informações que estão sendo protegidas pela autenticação de voz. Existem diversos tipos de ataques, os mais encontrados pela literatura podem ser visto a seguir [9]:

- Personificação da voz - Consiste em imitar o sinal de voz genuíno;
- Gravação da voz - Consiste na utilização de uma gravação do sinal de voz genuíno;
- Sintetização da voz - Consiste na sintetização de um sinal a fim de ser o mais parecido com o genuíno;
- Conversão de voz - Consistem em converter um sinal de voz a fim de ser o mais parecido com o genuíno.

Para o trabalho, foi escolhido a detecção de gravação, mesmo não sendo o ataque mais difícil de ser detectado, como é o ataque mais comum, principalmente devido a facilidade em que é possível realizar a gravação e a utilização de um áudio gravado, é importante que o sistema de autenticação consiga diferenciar um áudio gravado de um genuíno. Com isso, a utilização

de classificadores é uma opção interessante nessa área. Por exemplo, na edição de 2017 do *Automatic Speaker Verification Spoofing And Countermeasures Challenge* [10], o tema foi de ataques via *replay* de gravação de voz, o que mostra que a detecção desses ataques ainda é uma área promissora. Além disso, em 2019 um dos temas também foi ataques via *replay*.

## 1.4 Organização do trabalho

O trabalho está organizado em 5 capítulos, os quais estão explicados a seguir:

- No Capítulo 1, o leitor é introduzido na área de sistema de verificação de voz, a fim de entender a motivação desse trabalho e o quanto é importante;
- No Capítulo 2, é discorrido os conceitos e trabalhos publicados que são importantes para o entendimento do trabalho que foi desenvolvido;
- No Capítulo 3, é apresentado o desenvolvimento do trabalho em questão;
- No Capítulo 4, será apresentado os resultados que foram obtidos com os testes de classificação realizados;
- No Capítulo 5, a conclusão e a projeção de trabalhos futuros é apresentada.

## 1.5 Metodologia

Para o desenvolvimento desse trabalho, foi realizado um levantamento bibliográfico, a fim de obter conhecimento do tema e os trabalhos apresentados até o momento.

Posteriormente, a base de dados foi escolhida, foi utilizado os sinais disponibilizados pelo ASVspooF 2019 [11]. A partir desses sinais, foi realizado a conversão do formato FLAC para o

formato WAVE por meio das ferramentas Audacity [12] e fre:ac [13]. Com o formato WAVE, foi feita então a extração de características desse sinal. Para a criação dos vetores de características, foi utilizada a estimativa do espectro pela média de periodogramas [14] e *Teager Energy Operator* [15].

Por fim, os vetores de características foram utilizados para a classificação dos sinais por meio dos algoritmos de K-NN e SVM.

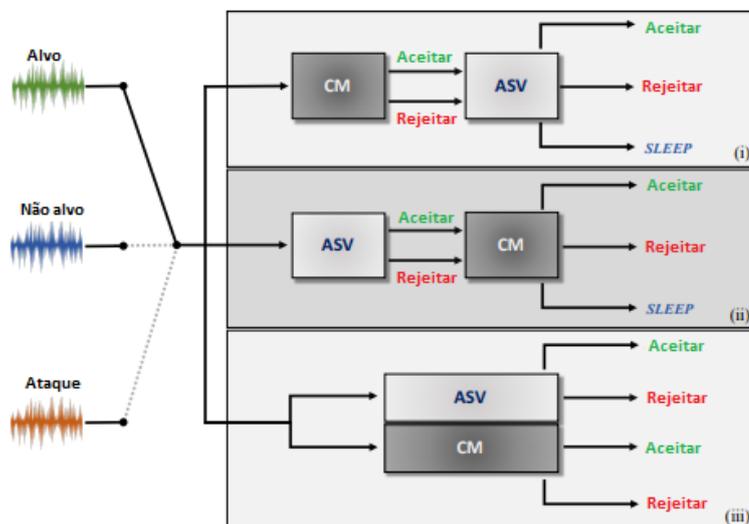
## Capítulo 2

### Revisão Bibliográfica

#### 2.1 Anti-falsificação de voz ou *Speech Spoofing Countermeasure*

O avanço de tecnologias como TTS, VC e microfones de alta qualidade se mostra uma grande ameaça para os ASV, mesmo com o uso de técnicas que se mostraram benéficas para a distinção de locutores, ainda não é suficiente para identificar ataques que usam essas tecnologias [9] [17]. Para contornar esse defeito do ASV, é utilizado um sistema de contra-medida em conjunto a esse autenticador, esse sistema é utilizado para classificar o sinal de voz, ou o aceita ou o recusa, esse sistema usualmente é chamada de CM (Countermeasure). Na figura 2.1, pode-se observar como é feita essa configuração da combinação entre os dois sistemas. Primeiramente, pode-se apresentar um modelo cascata, onde é utilizada um sistema atrás do outro, ou seja, caso o primeiro recuse não há a necessidade da verificação do sinal pelo segundo sistema, o que diminui o uso de recursos. Por último, a combinação dos sistema pode ser realizada em paralelo, onde ambos os sistemas realizam a autenticação e o sinal é aceito apenas se ambos os autenticadores aceitarem o sinal de voz.

**Figura 2.1** – Combinações possíveis de um CM e ASV. Em (i) e (ii) temos a apresentação do modelo cascata, onde os sistemas estão empregados um atrás do outro, CM e em sequência o ASV na primeira divisão, enquanto que, ASV e em sequência o CM na segunda divisão. Por último, temos que em (iii) a adoção do modelo paralelo.



Fonte: A imagem foi retirada e traduzida de: [18]

Os ataques mais comuns são realizados em dois ambientes de um ASV, são conhecidos como acesso físico (*Physical access*) e acesso lógico (*Logic access*). Esses dois ambientes, são reconhecidos como ataques diretos, os quais atacam o níveis do microfone e o de transmissão respectivamente. Além disso, temos a existência dos ataques indireto, os quais o foco é o próprio ASV, interferindo na extração das características, pontuação, entre outras partes do sistema de reconhecimento de padrões.

Os ataques diretos, em específico os ataques físicos os quais são foco desse trabalho, estão descritos a seguir:

- Acesso físico - O ataque a esse ambiente consiste na utilização do microfone, ou seja, o sinal de ataque é capturado pelo microfone do ASV. Usualmente os tipo de ataques realizados no acesso físico são a mimica e o "replay" de sinais de áudio gravados.
- Acesso lógico - O ataque a esse ambiente consiste na infiltração do sinal diretamente

no ASV usando a camada de transmissão, não tendo a utilização do microfone. Normalmente, os ataques que utilizam esse ambiente, são os ataques via sintetizadores e conversores de voz.

## 2.2 Formato Wave

O formato WAVE, conhecido como *Waveform Audio File Format*, é um formato de arquivo de áudio criado pela Microsoft em parceria com a IBM, bastante utilizado no processamento de sinais. O formato foi definido no RFC 2361 [21]. O formato, é uma variação do método RIFF, que é um arquivo de recipiente baseado em um armazenamento em *chunks* (blocos) [22].

A principal característica que torna esse formato bastante utilizado é que pode-se armazenar áudios compactados, mas, também armazenar áudios no formato PCM, que é um método de armazenamento não comprimido. Com isso, o formato é um ótimo meio de armazenamento para a análise de sinais.

## 2.3 Filtro de Pré-Ênfase

O filtro de pré-ênfase consiste em uma técnica de processamento de sinais que apresenta a finalidade de diminuir o efeito de ruído nos sinais a partir do aumento da amplitude de audio-freqüências altas e diminuição das menores. Na equação 2.1, é possível visualizar a aplicação do filtro em um sinal  $x$ , o que resulta em um novo sinal  $y$ .

$$y_i = x_i - 0.95x_{i+1} \quad (2.1)$$

## 2.4 Análise Espectral

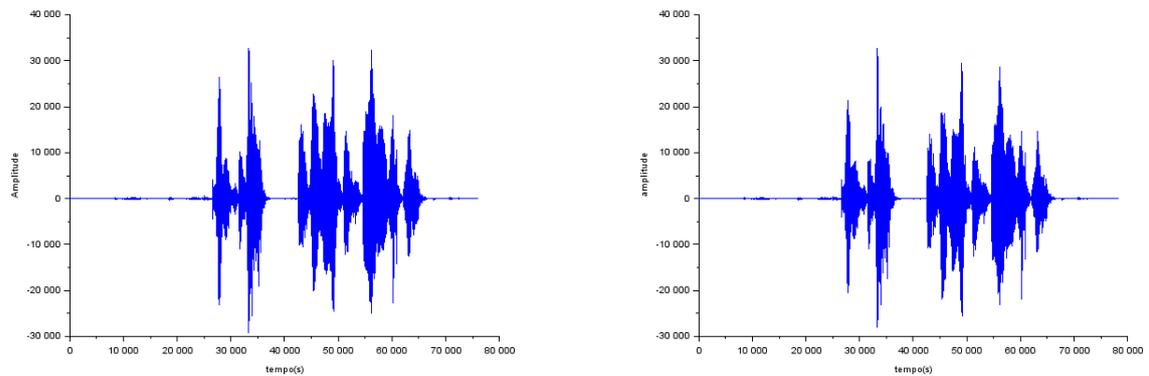
A forma usual de representação de um sinal de voz é uma função no domínio do tempo, a qual é a forma que é devolvida por um microfone. Entretanto, é muito mais fácil de analisar características que estão no domínio da frequência, devido a isso, usualmente é vantajoso trabalhar com o sinal neste domínio. A identificação e análise desses tipos de sinais é o que caracteriza a análise espectral.

### 2.4.1 Transformada discreta de fourier ou *Discrete Fourier Transform (DFT)*

A transformada discreta de fourier, consiste em uma técnica que leva um sinal que está no âmbito do tempo discreto para um que está no âmbito da frequência discreta. A DFT é muito importante para o processamento de sinais, devido a uma de suas propriedades de diminuir a quantidade de dados sem apresentar uma perda do sinal, podendo escolher o seu tamanho, ou seja, o número de pontos que a DFT irá apresentar. Olhando um sinal de voz gravado, ele está no âmbito do tempo, devido a isso, o sinal apresenta diversos pontos, o que diminui o desempenho e dificulta a análise desse sinal em questão. Com isso, podemos aplicar a DFT nesse sinal para possibilitar sua análise.

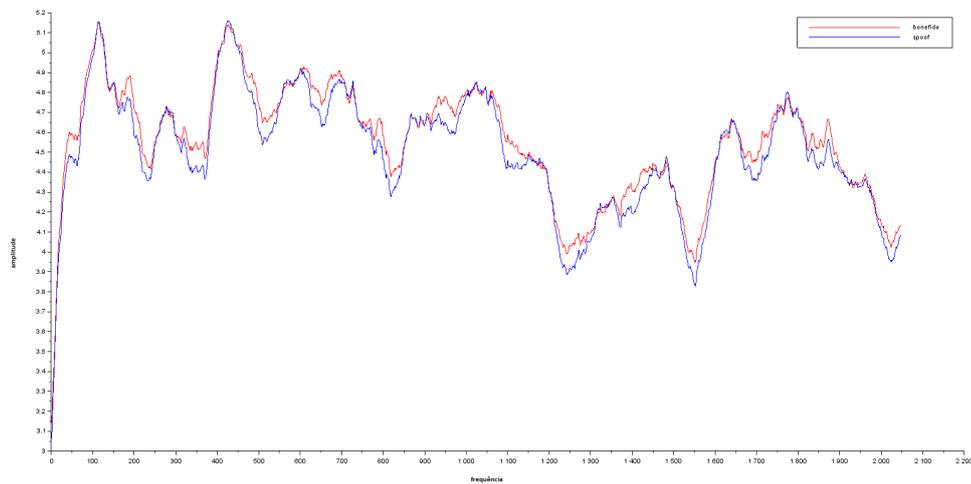
Por exemplo, no conjunto de figuras 2.2, podemos ver um sinal na dimensão do tempo e o mesmo na dimensão da frequência. Vemos que, temos um número de menor de pontos para a DFT, mas ainda conseguimos analisar o sinal e com mais eficiência. Comparando com a gravação desse sinal, é possível notar diferença entre o sinal gravado e o verdadeiro apenas analisando o gráfico, o que mostra que a DFT apresenta propriedades que podem ser úteis na detecção de *spoofings* (ataques).

**Figura 2.2** – Visualização da DFT.



(a) Sinal de voz.

(b) Replay do sinal em (a).



(c) Comparação da DFT de (a) e (b).

Fonte: Confeccionado pelo autor.

A transformação consiste em decompor o sinal em senoides tendo sua saída a amplitude da frequência desses senoides. Para exemplificar, na equação 2.2 temos  $X[k]$  o sinal de saída,  $x$  o sinal de entrada,  $n$  o número de pontos do sinal de entrada e  $N$  é o tamanho da DFT.

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j \frac{2\pi}{N} kn} \quad (2.2)$$

## 2.5 Extração de Características

A extração de características é uma fase importante para o reconhecimento de padrões. A técnica consiste em, a partir de um conjunto inicial de dados medidos, transformá-los em valores informativos (características). Esses novos valores, ou vetor de características, apresentam um tamanho fixo e são utilizados para reconhecer padrões e assim diferenciar dois ou mais sinais.

Nesta subseção, será mostrado algumas técnicas de extração que foram utilizadas nesse trabalho, as quais podem ser vistas na seção 2.5.1.

### 2.5.1 Estimativa da densidade do Espectro

Ao utilizar apenas um pedaço do sinal e aplica-lo uma transformada, chamado periodograma, pode se estimar a densidade do espectro do sinal, entretanto, não é consistente[14]. Para diminuir as variações presentes nessa estimativa, é utilizado métodos existentes para isso, como o apresentado a seguir, o Método de Welch.

#### Método de Welch

O método de Welch consiste em calcular os periodogramas e obter sua média a fim de melhorar a estimativa da densidade do espectro. A seguir esta listado o procedimento para realizar o método:

1. Dado um sinal de  $M$  pontos, ele será dividido em  $Q$  segmentos de comprimento  $N$ , onde  $Q$  não é a divisão correta de  $M/N$ . Além disso, os segmentos podem apresentar sobreposição, ou seja, pontos presentes em ambos os segmentos.
2. Posteriormente é aplicado uma função janela.
3. Após a aplicação da função, deve-se realizar o cálculo da DFT de cada janela.

4. Com o cálculo da DFT, devemos obter a sua magnitude aplicando seu conjugado.
5. Por fim, deve-se realizar a media das janelas.

Os passos 1,2 e 3 são aplicados a partir da equação 2.3. Para encontrar a magnitude, passo 4, basta multiplicar a DFT pelo seu conjugado, como está na equação 2.4. No último passo, é feita média das janelas como é representado na equação 2.5

$$X[k] = \frac{1}{N} \sum x[n] e^{-j\frac{2\pi}{N}kn} \quad (2.3)$$

$$J_{xx} = X[k] \cdot X^*[k] \quad (2.4)$$

$$S_{xx} = \frac{1}{Q} \sum_{q=1}^Q J_q[k] \quad (2.5)$$

### 2.5.2 Teager Energy Operator (TEO)

Para gerar um sinal de determinada frequência, é necessário uma certa quantidade de energia, sendo que, essa energia pode ser calculada a partir do sinal no âmbito da frequência discreta (DFT) visto na 2.4.1. A TEO, é bastante utilizada no processamento de sinais e apresenta avanços em uma das suas sub-áreas [15], o processamento de voz.

O cálculo da da TEO é bem simples, ela parte do principio das leis de movimento de Newton, onde se pegamos um objeto de massa  $m$  conectado a uma mola e aplicar uma diferencial de segunda ordem, obtemos a equação 2.6. Como a variável  $x$  é a posição do objeto em determinado momento  $t$ , se o objeto é um sinal, então,  $x = A \cdot \cos(\omega \cdot t + \phi)$  (movimento harmônico), logo, a solução para equação se da quando a frequência angular  $\omega = (\frac{k}{m})^{\frac{1}{2}}$ , para qualquer amplitude  $A$  e fase  $\phi$ .

$$\frac{\partial^2 x}{\partial t^2} + \frac{k}{m}x = 0 \quad (2.6)$$

Devido a energia mecânica do sistema ser a soma da energia potencial da mola e a energia cinética, é possível calcular a energia do sinal a partir da formula 2.12 ao substituir  $x$  na equação

2.8.

$$E = E_p + E_c \quad (2.7)$$

$$E = \frac{kx^2}{2} + \frac{mv^2}{2} \quad (2.8)$$

$$v = \frac{\partial x}{\partial t} \quad (2.9)$$

$$E = \frac{A^2 \cdot k}{2} \quad (2.10)$$

$$k = \omega^2 \cdot m \quad (2.11)$$

$$E = \frac{1}{2} A^2 \omega^2 m \quad (2.12)$$

Se a massa se manter constante, a energia é apenas dependente de A e  $\omega$ , ou seja, proporcional a elas, logo:

$$E \propto A^2 \omega^2 \quad (2.13)$$

Utilizando o algoritmo de Kaiser [16], onde a energia de uma amostra do sinal de frequência discreta é calculada através da equação 2.14.

$$x_n = A \cdot \cos(\Omega \cdot n + \phi) \quad (2.14)$$

$$\Omega = \frac{\omega}{2\pi} \quad (2.15)$$

Calcular a energia do sinal pode ser feita utilizando as amostras do sinal em questão.

$$x_{n-1} = A \cdot \cos(\Omega) \cdot (n - 1) + \phi \quad (2.16)$$

$$x_n = A \cdot \cos(\Omega) \cdot n + \phi \quad (2.17)$$

$$x_{n+1} = A \cdot \cos(\Omega) \cdot (n + 1) + \phi \quad (2.18)$$

Como

$$x_{n-1} \cdot x_{n+1} = A \cdot \cos(\Omega) \cdot (n - 1) + \phi \cdot A \cdot \cos(\Omega) \cdot (n + 1) + \phi \quad (2.19)$$

Ao utilizar a identidade trigonométrica abaixo:

$$\cos(\alpha - \beta) \cdot \cos(\alpha + \beta) = \frac{\cos(2\alpha) + \cos(2\beta)}{2} \quad (2.20)$$

Temos que:

$$x_{n-1} \cdot x_{n+1} = \frac{A^2 \cos(2\Omega n + 2\phi) + \cos(2\Omega)}{2} \quad (2.21)$$

Utilizando outra identidade trigonométrica presente na equação 2.22, o cálculo de  $x_{n-1} \cdot x_{n+1}$  pode ser visto na equação 2.26

$$\cos(2\alpha) = 2 \cos^2(\alpha) - 1 = 1 - 2 \operatorname{sen}^2(\alpha) \quad (2.22)$$

$$x_{n-1} \cdot x_{n+1} = \frac{A^2(2 \cos^2(\Omega n + \phi) - 1 + 1 - 2 \operatorname{sen}^2(\Omega))}{2} \quad (2.23)$$

$$x_{n-1} \cdot x_{n+1} = A^2 \cos^2(\Omega n + \phi) - \operatorname{sen}^2(\Omega) \quad (2.24)$$

$$x_n = A \cdot \cos(\Omega) \cdot n + \phi \quad (2.25)$$

$$x_{n-1} \cdot x_{n+1} = x_n^2 - A^2 \operatorname{sen}^2(\Omega) \quad (2.26)$$

Para valores pequenos de  $\Omega$ ,  $\operatorname{sen}(\Omega)$  pode ser aproximado para  $\Omega$ . Logo, a partir da equação 2.26, a aproximação da energia pode ser encontrada através da equação 2.28

$$A^2 \operatorname{sen}^2(\Omega) = x_n^2 - x_{n-1} \cdot x_{n+1} \quad (2.27)$$

$$A^2 \Omega^2 \propto x_n^2 - x_{n-1} \cdot x_{n+1} = T \quad (2.28)$$

Com isso, obtemos a *Teager Energy Operator* ou *Teager Kaiser Energy Operator*, representada como T.

## 2.6 Validação cruzada

A validação cruzada, mais conhecida como *Cross Validation*, é uma técnica de *machine learning* comumente utilizada para verificar a capacidade de generalização de um modelo de classificação [19]. A técnica consiste em, particionar a base de dados em subconjuntos exclusivos, um de treinamento e outro de validação, a fim de utiliza-los para estimar a acurácia do modelo. O subconjunto de treinamento é utilizado para treinar o classificador, enquanto que, o subconjunto de validação será classificado pelo classificador.

Primeiramente, é necessário verificar como será realizado o particionamento. Para isso, os principais métodos são [20]:

- *Holdout* - Divisão do conjunto de dados em dois subconjuntos mutuamente exclusivos. Usualmente, a divisão adotada é de 2/3 do data set é direcionado para o treinamento, enquanto que, 1/3 é para o conjunto de testes.
- *Leave-one-out* - Consiste em dividir o conjunto de dados em k subconjuntos mutuamente exclusivos, onde k é o número total de dados. Com isso, realiza-se k cálculos de erros.

Para calcular a acurácia de um teste, é comumente utilizado uma matriz de confusão a qual pode ser vista na matriz na equação 2.29, onde as colunas representam o valor dado pelo classificador e a linha o valor real, ou vice-versa. De maneira simples, podemos calcular a matriz da seguinte forma:

- Adiciona-se 1 em Verdadeiro Positivo caso o classificador faça uma previsão precisa.
- Adiciona-se 1 em Falso Negativo caso o classificador classifique como falso uma verdade.
- Adiciona-se 1 em Falso Positivo caso o classificador classifique como verdade uma falsidade.
- Adiciona-se 1 em Verdadeiro Negativo caso o classificador classifique como falso uma falsidade.

$$Matriz = \begin{bmatrix} \text{Verdadeiro Positivo} & \text{Falso Negativo} \\ \text{Falso Positivo} & \text{Verdadeiro Positivo} \end{bmatrix} \quad (2.29)$$

Com a matriz de confusão, pode-se então, estimar a acurácia da partição. A acurácia, é calculada a partir da soma dos Verdadeiros Positivos obtidos com os Verdadeiros Negativos obtidos, dividindo a soma

pela população total, ou seja, a quantidade total de vetores de características utilizados, como pode ser visto na equação 2.30.

$$ACC = \frac{\sum \text{Verdeiro Positivo} + \sum \text{Verdadeiro Negativo}}{\text{População Total}} \quad (2.30)$$

Uma outra forma de verificar um modelo, é através da medida Equal Error Rate (EER). A EER, é um ponto onde a taxa de falso negativo é igual a taxa de falso positivo, seu valor é dado pela taxa de falso negativo ou taxa de falso positivo, quando ambos os valores são iguais. Entretanto, caso não haja taxas iguais, a EER se dá pela média de ambas as taxas com menor diferença, como pode ser visto na equação 2.31, onde  $fpr$  é a taxa de falso positivo (false positive rate) e  $fnr$  é a taxa de falso negativo (false negative rate).

$$EER = (fpr + fnr)/2 \quad (2.31)$$

## 2.7 Classificação

Na área do aprendizado de máquina, temos uma sub-área conhecida como classificação. A classificação é uma técnica que visa atribuir a um dado que está sendo analisado uma classe, a classe é uma maneira de identificar esse dado a um conjunto específico [23] [24]. Para esse trabalho em questão, temos duas classes: sinal falsificado (*spoof*) e sinal genuíno (*bonafide*). Com isso, o trabalho visa utilizar um classificador para classificar sinais de voz nessas duas classes.

Em primeiro lugar, é realizado um treinamento no classificador a partir de dados que foram escolhidos para essa finalidade, normalmente esse conjunto é criado a partir de uma validação cruzada, os quais serão utilizados por um algoritmo de treinamento. O treinamento, pode ser supervisionado, o qual o modelo apresenta resultados pré-definidos, ou, não supervisionado, onde o modelo não utiliza resultados pré-definidos.

Em segundo lugar, é realizado um teste com o modelo treinado com um novo conjunto de dados, o conjunto de validação. Nessa etapa, o segundo conjunto é classificado a partir das informações que o modelo adquiriu do treinamento.

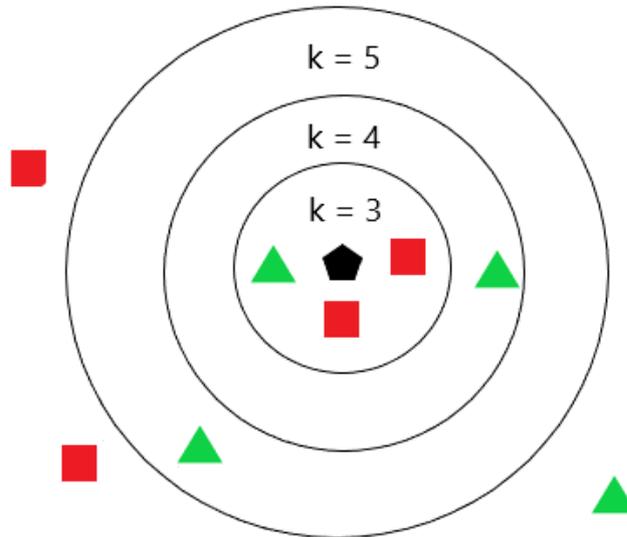
Por último, é realizada a validação do modelo, onde é verificado se o modelo treinado atual é apto para classificar dados que não estão presentes nessas etapas da classificação, ou seja, se o modelo está generalizado. Caso isso não ocorra, o modelo irá refazer o treinamento, se após muitas iterações o classificador não obteve uma boa pontuação nas medidas utilizadas para verificação da generalização do modelo, pode ser necessário mudar os vetores de características ou o próprio classificador.

### 2.7.1 k-vizinhos mais próximos ou k-Nearest Neighbors (k-NN)

Um dos principais algoritmos utilizados para classificação é o k-vizinhos mais próximos, proposto em 1951 [28], a sua simplicidade em conjunto a sua performance que consegue competir com classificadores mais complexos e atuais, o torna umas das ferramentas mais comuns no meio do *machine learning* [27].

A técnica é classificar um ponto baseado nos k pontos mais próximos. Para exemplificar, temos um ponto que queremos classificar como falso ou verdadeiro, as nossas classes, a partir de um conjunto de pontos, que no *machine learning* seria o conjunto de dados de treinamento, a classificação irá se basear na proximidade desses pontos de treino com o ponto a ser classificado, baseando-se em uma métrica de distância, como por exemplo, a Euclidiana, Manhattan, Minkowski, entre outras [25] [29]. Logo, para cada ponto a ser classificado iremos verificar o rótulo dos k pontos na suas proximidades, e a classe com maior quantidade nessa proximidade é a classe do ponto a ser classificado [26]. Na figura 2.3, há uma exemplificação de um k-NN com  $5 \leq k \leq 3$  e com uso da distância euclidiana. Percebe-se que, a classificação pode mudar dependendo do k escolhido, sendo importante então, encontrar k ótimo para a solução. Uma maneira simples de aplicar o k-NN pode ser vista no algoritmo 1 apresentado em [27], o qual foi traduzido para este trabalho.

**Figura 2.3** – Exemplo de uma classifica de k-NN com  $k = 3$ ,  $k = 4$  e  $k = 5$ .



Fonte: Confeccionado pelo Autor.

---

**Algoritmo 1:** Algoritmo básico de K-NN

---

**Entrada:** Amostra de teste  $d$ , Conjunto de treino  $D$  e quantidade de amostras próximas  $k$

**Saída:** Classificação da amostra de teste  $d$

- 1 - Calcular a distancia entre  $d$  e todas as amostras presentes em  $D$
  - 2 - Pegar as  $k$  amostras mais próximas de  $d$
  - 3 - Verificar a classe mais frequente entre as  $k$  amostras e atribui-la a amostra de teste  $d$
- 

### Distância Euclidiana

Uma das métricas de distâncias utilizadas no classificador k-NN é a distancia euclidiana. A métrica consiste no segmento de linha entre dois pontos presentes em um espaço euclidiano. Com isso, para calcular o tamanho do segmento entre dois pontos  $P$  e  $Q$ , por meio de suas coordenadas cartesianas e do teorema de Pitágoras, é obtida a distância entre eles. Na equação 2.32, pode-se observar o cálculo da

distância entre dois pontos em um espaço euclidiano de  $n$  dimensões.

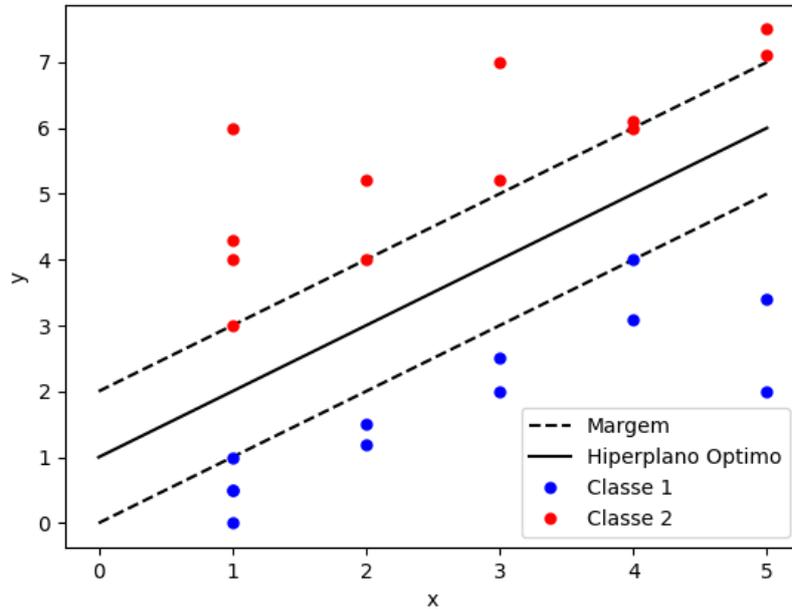
$$d(P, Q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2 + \dots + (p_{n-1} - q_{n-1})^2 + (p_n - q_n)^2} \quad (2.32)$$

## 2.7.2 Máquina de Vetores de Suporte ou Support Vector Machine (SVM)

No meio dos algoritmos de classificação via aprendizado supervisionado, temos além do k-NN a técnica da Máquina de Vetores de Suporte, comumente chamada de Support Vector Machine (SVM). Nesta técnica, o objetivo é encontrar um hiperplano que consiga dividir dois conjuntos de exemplos, a partir de suas características, em duas classes distintas [25].

O hiperplano, de maneira simples, é uma linha que separa um conjunto de dados em duas classes sendo um limite de decisão do classificador, para uma visualização vemos a separação por hiperplano na figura 2.4. Para obter o hiperplano ótimo, basta encontrar o hiperplano com a maior margem possível a partir das observações utilizadas, onde a margem é calculada como sendo a distância mínima das observações de cada classe. Entretanto, na maioria dos casos não temos limites de classe lineares, para isso, é necessário a utilização dos *kernels*. O *kernels*, é uma forma de adequar o limite de classe para a dimensão adequada a partir de uma função linear, polinomial ou radial.

**Figura 2.4** – Exemplo de divisão de classes via Hiperplano.

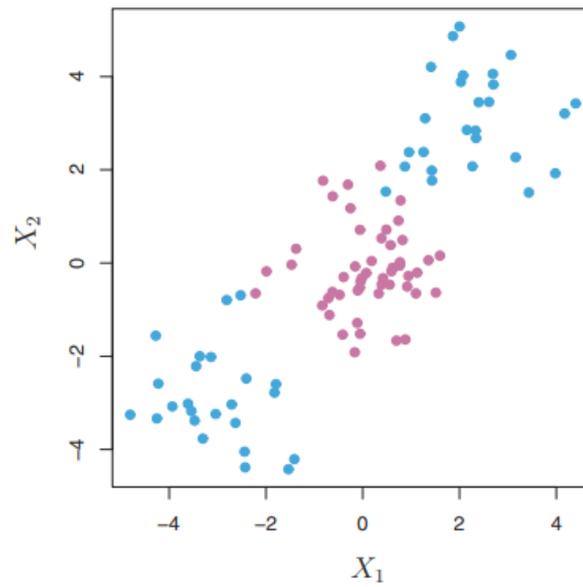


Fonte: Confeccionado pelo Autor.

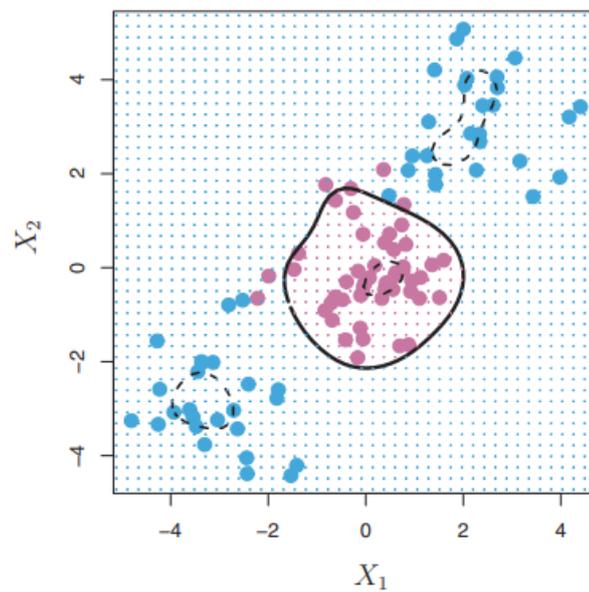
Para esse trabalho, será utilizado o *Kernel* Guassiano, conhecido como conhecido *radial basis function* (RBF). O RBF, é uma forma de visualizar a distancia entre dois pontos de dimensão do tamanho dos vetores de características. Na equação 2.33, temos o cálculo do *kernel* a partir da distancia de  $v_1$  e  $v_2$ , onde  $v_1$  e  $v_2$  são os vetores de características e  $\sigma$  a variância, ou fator de dimensionamento, a qual é um valor arbitrário. A ideia geral do RBF, é ignorar os exemplos de treino que estão muito longe, isso pode ser visto ao utilizar a função exponencial na distancia, onde valores altos tornam-se pequenos, não interferindo na predição da classe. Na figura 2.5(a), é visualizado um conjunto linearmente inseparável, ou seja, não é possível dividi-lo de maneira eficiente com uma reta, já na figura 2.5(b), a construção do hiperplano a partir de um SVM *kernel* RBF é realizada, tornando a separação possível. Além disso, nota-se que as distancia altas são ignoradas e a divisão é bem localizada.

$$[H]K(v_1, v_2) = \exp\left(-\frac{\|v_1 - v_2\|^2}{2\sigma}\right) \quad (2.33)$$

**Figura 2.5** – Aplicação de *kernel* em um conjunto de características linearmente inseparáveis.



(a) Conjunto de observações linearmente inseparáveis.



(b) Divisão após aplicação de uma SVM com *kernel* RBF

Fonte: Retirado de JAMES et al., 2013 [30].

## 2.8 Trabalhos relacionados

Nesta seção, são apresentados alguns trabalhos da área detecção de ataque à ASV. O objetivo da apresentação desses trabalhos, é mostrar os principais focos da área, expondo assim, o estado da arte.

No artigo de Witkowski *et al* [1], os autores propõem a exploração das alterações na amplitude no âmbito da frequência que os ataques via *replay* apresentam quando comparados aos sinais originais e como isso pode ajudar no sistema base de ASV apresentado no evento ASVspoof 2017 *challenge*. Para isso, foram utilizados as seguintes características do âmbito da frequência: *Constant Q Cepstral Coefficients* (CQCC), *Cepstrum*, *Mel-Frequency Cepstral Coefficients* (MFCC), *Inverse Mel-Frequency Cepstral Coefficients* (IMFCC), *Linear Prediction Cepstral Coefficients* (LPCC), *Linear Prediction Cepstral Coefficients Residual Part* (LPCRes). Cada característica, foi aplicada em um classificador utilizando a base de dados disponibilizada pelo ASVspoof 2017, para realizar a classificação os autores propuseram o uso de dois modelos gaussianos de mistura (GMM) com um treinamento de maximização de expectativa. Foram obtidas, então, as taxas de erros *Equal Error Rate* (EER), sendo as menores para cada característica, 5.13% (CQCC), 3.38% (*Cepstrum*), 3.16% (MFCC), 16.18% (IMFCC) e 6.22% (LPCRes), onde, as duas primeiras taxas estão na faixa de banda de 6000-8000hz e as 3 últimas na faixa de banda de 4000-8000hz.

No trabalho de Himawan *et al* [2], é exposto a necessidade da criação de contra-medidas de ataques a ASV. Os autores, apresentam diversos testes para as áreas de detecção de ataques, como a síntese de voz, conversão de voz e ataques via *replay*. Na área de ataques via *replay*, foi aplicado um rede neural convolucional (CNN) para realizar a classificação a partir dos data sets BTAS 2016 e ASVspoof 2015. A menor taxa de erro, foi apresentada quando houve a junção de ambos os data sets, observando uma taxa de EER de 0.69%

Em Tak *et al* [3], é proposto a utilização *Linear Frequency Residual Cepstral Coefficients* (LFRCC) usufruindo de técnicas de pós-processamento e *Linear Triangular Filterbank*. No trabalho, os autores tiveram como data set o ASVspoof 2017, para serem utilizado por dois classificadores GMM e CNN obtendo uma taxa EER de 2.40% e 9.06% no conjunto de desenvolvimento e validação respectivamente.

Kamble *et al* [4], continuação de seu outro trabalho [5], aplicaram o uso do *Teager Energy Operator* (TEO), utilizando as sub-bandas de energia para encontrar o *Teager Energy Cepstral Coefficients*

(TECC). Para realizar a classificação, os autores utilizaram um classificador GMM e o data set ASVspoof 2017 V2.0, realizando testes com diversas técnicas já reconhecidas, CQCC, *Linear Frequency Cepstral Coefficients* (LFCC), MFCC e o método proposto TECC. No final, a menor taxa de erro EER foi encontrado com 6.68% e 10.45% para os conjuntos de desenvolvimento e validação respectivamente.

Javed *et al* [6], propuseram um modelo de contra-medida para sistemas de controle de voz, com o classificador SVM alimentado pela feature *Acoustics Ternary Pattern* (ATP) em conjunto à característica *Gammatone Cepstral Coefficients* (GTCC), utilizando a base disponibilizada no ASVspoof 2019. Como resultado, foi obtido uma taxa de erro EER de 1% para ataques via *replay*.

Neste trabalho, o objetivo é o mesmo dos artigos apresentados, onde é apresentado um método para a detecção de ataques via *replay*, usufruindo de técnicas de *machine learning*. Para esse projeto, optou-se pelo uso dos classificadores k-NN e SVM para classificar um sinal de voz a partir das características apresentadas nesse trabalho. Consequentemente, cada classificador teve sua capacidade averiguada a partir de sua acurácia e EER, para, no fim, verificar qual classificador é mais apto a trabalhar com sistemas de contra-medidas.

# Capítulo 3

## Detalhamento do Trabalho Proposto

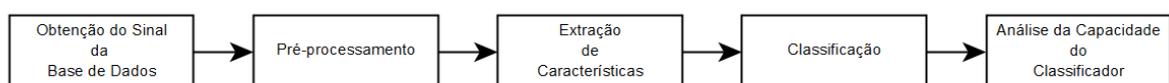
### 3.1 Considerações iniciais

Nesse capítulo, é apresentado o processo de desenvolvimento do trabalho e detalhamento de cada etapa apresentada na seção 1.5 do capítulo 1, a partir dos conceitos que foram apresentados no capítulo 2. O desenvolvimento desse trabalho, consiste em criar um classificador que a partir de um sinal de voz consiga classificar com uma boa acurácia se a classe do sinal de voz é uma ataque via *replay* ou se é um sinal de voz genuíno. O fluxo do trabalho pode ser visto na figura 3.1

### 3.2 Base de Dados

Os sinais utilizados para esse trabalho, foram obtidas a partir da base de dados construída pela *Automatic Speaker Verification and Spoofing Countermeasures Challenge* [10] a qual foi disponibilizada

**Figura 3.1** – O fluxo do desenvolvimento do trabalho.



Fonte: Confeccionado pelo autor.

em seu terceiro evento, ASVspoof 2019 [11]. Para o trabalho, utilizou-se 4 conjuntos de sinais de ataques, com diferentes qualidades do dispositivo de reprodução, cada conjunto com 50 sinais de ataques aleatórios para serem utilizadas como base do desenvolvimento do trabalho e outros 50 sinais genuínos. Além disso, foram utilizados os mesmos 50 sinais genuínos para todos os conjuntos.

A base de dados foi dividida entre sinais de acesso físico e acesso lógico, disponíveis no formato .flac, para o trabalho foram utilizado os sinais de acesso físico. A identificação de um sinal de voz na base de dados é atribuída a 4 identificadores, para sinais genuínos, e, 5 identificadores para os ataques, como pode ser visto a seguir:

1. Um identificador de 4 dígitos do locutor.
2. Nome do arquivo de áudio.
3. Uma tripla de 3 letras, ver tabela 3.1
4. Um dupla para identificar as configurações do ataque, ver tabela 3.2
5. Identificador da classe do sinal, *bonafide* para genuíno e *spoof* para os sinais utilizados nos ataques.

**Tabela 3.1** – Tabela de identificação de características do sinal.

	a	b	c
Tamanho do Quarto ( $m^2$ )	2.5 - 5	5 - 10	10 - 20
T60 (ms)	50 - 200	200 - 600	600 - 1000
Distancia até o ASV (cm)	10 - 50	50 - 100	100 - 150

**Tabela 3.2** – Tabela de identificação de características do ataque.

	a	b	c
Distância do ataque para o ASV (cm) ( $m^2$ )	10 - 50	50 - 100	maior que 100
Qualidade do dispositivo de <i>replay</i>	perfeita	alta	baixa

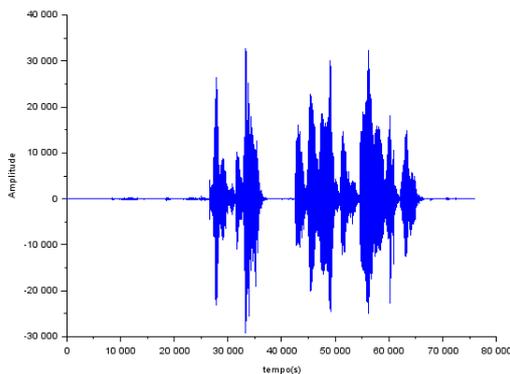
Com a obtenção da base de dados, foram criados 4 conjuntos com 100 arquivos de áudio, cada conjunto com qualidades diferentes de dispositivos de reprodução, qualidade perfeita (QP), qualidade alta (QA), qualidade baixa (QB) e todas as qualidades. Os áudios escolhidos foram convertidos do formato original comprimidos, .flac, para o formato WAVE, o qual não há a compressão de dados. A conversão foi feita por meio do programa fre:ac v.1.1.2a [13]. Foram selecionados, uma parcela dos áudios convertidos no formato WAVE, para serem comparados com os arquivos .flac através do software livre Audacity versão 2.4.1 [12] para verificar nenhuma perda de informação na conversão.

### 3.3 Pré-processamento

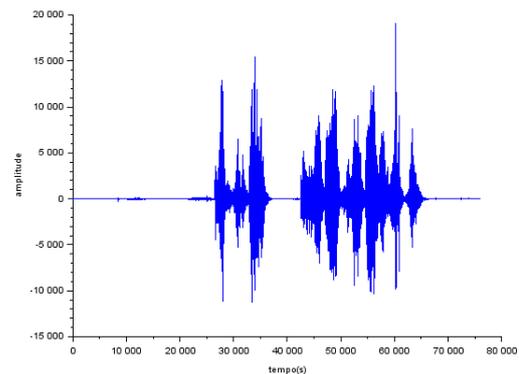
Com os dados convertidos, foi utilizada uma função escrita na linguagem C/C++ para fazer a retirada das amplitudes do sinal presente no arquivo WAVE, as quais, foram armazenadas em arquivos no formato .txt. Na figura 3.2(a), é possível observar uma representação gráfica das amplitudes obtidas por meio da função. Além disso, após a retirada das amplitudes, foi aplicado o filtro de Pré-Ênfase, filtro visto na seção 2.3, nos pontos do sinal a partir da equação presente em 3.1, o sinal produzido a partir da Pré-Ênfase pode ser visto em 3.2(b).

$$y_i = x_i - 0.95x_{i+1} \quad (3.1)$$

**Figura 3.2** – Aplicação do filtro de Pré-Ênfase.



(a) Sinal de voz da base de dados.



(b) Filtro de Pré-Ênfase aplicado em (a).

Fonte: Confeccionado pelo autor.

### 3.4 Extração de características

Para prosseguirmos para a etapa de classificação, temos que realizar um procedimento tao importante quanto, a construção dos vetores de características. Serão extraídas, dois tipos de características, a energia (TEO) e a estimativa da densidade do espectro, as quais serão comparadas no final desse trabalho.

Primeiramente, foi aplicado nos sinais produzidos pelo pré-processamento o método de Welch, o qual foi explicado na seção 2.5.1. Entretanto, houveram algumas mudanças no método original, os procedimentos estão descritos a seguir:

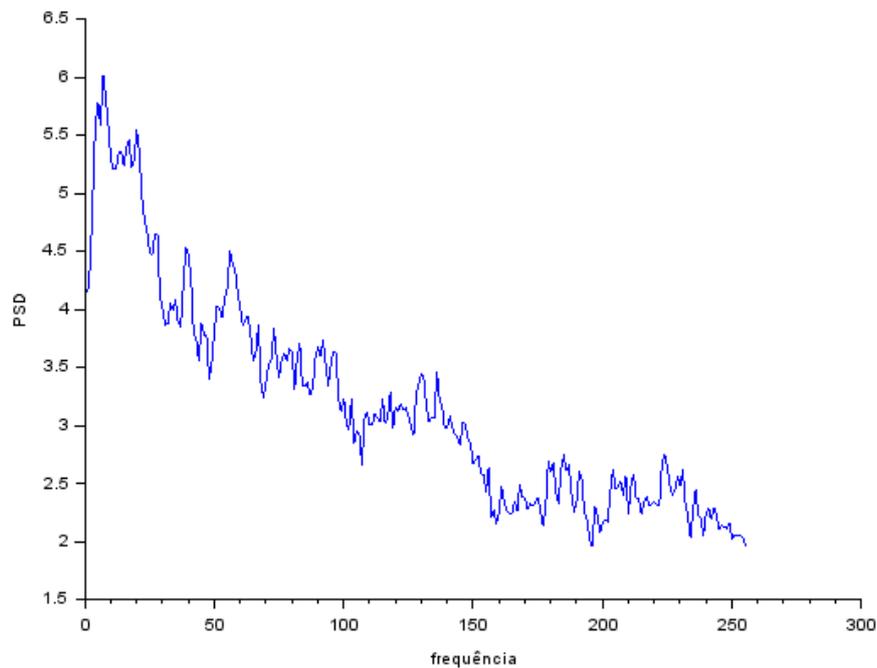
- Foi realizada a divisão do sinal em Q partes com espaçamento de 256 pontos, diferente do usual, a qual é feito uma sobreposição de janelas.
- O tamanho da janela escolhida é de  $N = 512$  pontos.
- Não foi utilizado uma função janela.
- Após a construção das janelas, foi realizada a DFT de cada janela em seu tamanho original, ou seja, 512 pontos.
- Por fim, foi feita a media das transformadas realizadas.

Os passos 1, 2, 3 e 4 são visualizados na equação 3.2.

$$X[k] = \frac{1}{N} \sum x[n] e^{-j\frac{2\pi}{N}kn} \quad (3.2)$$

Na figura 3.3, temos um exemplo visual da estimativa do espectro.

**Figura 3.3** – Estimativa da densidade do espectro utilizando uma janela de 512 pontos.



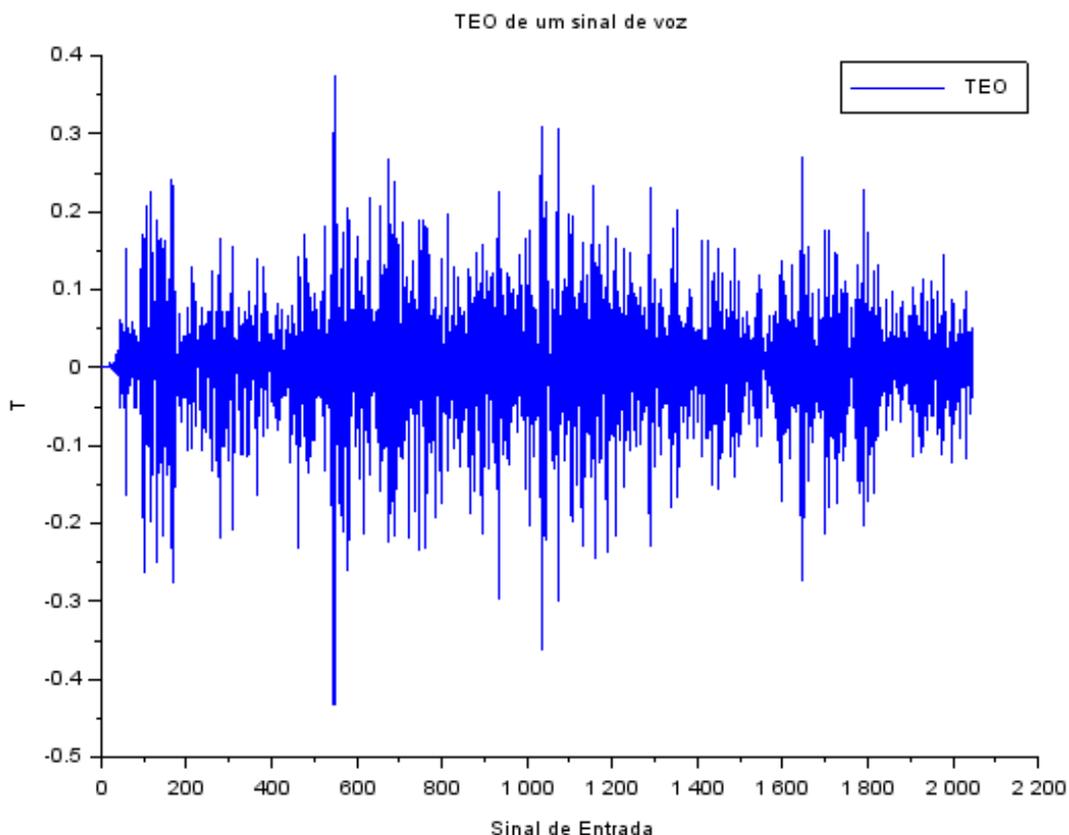
Fonte: Confeccionado pelo autor.

Por último, foi realizado também a extração da energia (TEO) dos sinais do pré-processamento, construindo-se um novo conjunto de vetores de características. Para a estimação da TEO, o tamanho da DFT foi de 4096, totalizando então um vetor de características de 2046 pontos. O cálculo do vetor pode ser visto na equação 3.3.

$$T[n] = x[n]^2 - x[n-1] \cdot x[n+1], 0 < n < 2048 \quad (3.3)$$

Na figura 3.4, é possível ver um exemplo da TEO.

**Figura 3.4** – *Teager Energy Operator* de um sinal de entrada de 2048 pontos.



Fonte: Confeccionado pelo autor.

Com isso, temos então, os vetores de características construídos. Cada vetor, será utilizada para o treinamento e validação do classificador utilizado, o qual pode ser visto na seção 3.5.

## 3.5 Classificação

Nessa etapa, a base de dados, criada na extração, utilizada para o treinamento foi dividida em 2 conjuntos iguais para treinamento e validação, ou seja, 50 vetores para realizar o treinamento do classificador e 50 para a validação. Além disso, foram realizadas iterações da validação cruzada com combinações distintas de conjuntos de entrada para os classificadores utilizados. Ademais, 4 tipos de conjuntos foram criados, para cada qualidade de dispositivo de replay, ou seja, 3 tipos para as qualidades perfeita, alta e

baixa, e, 1 tipo para todas as qualidades em conjunto.

Um dos classificadores é o modelo k-NN, discutido na seção 2.7.1. Para calcular a distância dos vetores de características, foi utilizada a versão do k-NN com a distância Euclidiana, definida na equação 3.4, onde  $v_1$  e  $v_2$  são os vetores de características de treino e teste respectivamente e N sendo o número de pontos de cada vetor.

$$d(v_1, v_2) = \sqrt{\sum_{i=1}^N (v_1(i) - v_2(i))^2} \quad (3.4)$$

O outro modelo testado é o SVM, o qual foi discutido na seção 2.7.2. Para adequar a dimensão do problema, foi utilizado o *kernel* mais comum, a função de base radial, ou *radial basis function* (RBF), do tipo Gaussiano, definido na equação 3.5, onde valores  $v_1, v_2$  são os exemplos de treinamento do modelo e a variância  $\sigma$  tem para este trabalho valor 1 .

$$K(v_1, v_2) = \exp\left(-\frac{\|v_1 - v_2\|^2}{2\sigma}\right) \quad (3.5)$$

Por fim, ambos os classificadores irão calcular a acurácia (ACC) e equal error rate (EER) da validação cruzada e a retornará. Além disso, a matriz de confusão do melhor valor de ACC é armazenada e amostrada pelo classificador. A partir disso, é realizado uma análise das ACCs e EERs encontrados nos testes realizados no capítulo 4, para verificar se o classificador está apto a realizar a classificação dos sinais de voz.

# Capítulo 4

## Testes e Resultados

### 4.1 Pontos Importantes

Para todos os testes, foram utilizados no total 100 exemplos de sinais, para cada conjunto de qualidade, divididos em 50%, ou seja, 50 exemplos de sinais verdadeiros e 50 exemplos de sinais de ataque. Além disso, foi realizados 1000 iterações de validação cruzada para, assim, obter uma média de acurácia e EER significantes.

### 4.2 Testes k-NN

No primeiro teste do classificador k-NN, foram utilizados apenas exemplos de ataques de qualidade de dispositivo perfeita. Os resultados obtidos pelos vetores de características, TEO e Janela, foram bem parecidas, como pode ser visto na tabela 4.1, com uma média de acurácia de 54,55% para o janelamento e 50,06% para a TEO.

No segundo teste do classificador k-NN, foram utilizados apenas exemplos de ataques de qualidade de dispositivo altas. A diminuição da qualidade, trouxe uma diferença na taxa de erro e acurácia obtidas no teste anterior, com um aumento de 9,37% e 9,66% na acurácia e uma diminuição brusca da EER, com

destaque ao janelamento que obteve uma EER de 0.26 vista na sétima linha da tabela 4.1. Com isso, percebe-se que o modelo não possui tanta eficiência para ataques de qualidade perfeita e é afetado pela qualidade do dispositivo de ataque.

No penúltimo teste do k-nn, no qual foi utilizado ataques de qualidade baixa, houve um aumento considerável na capacidade de generalização do modelo. Em relação ao janelamento, o modelo teve uma média de acurácias de 87,76%, um aumento de 23,84% quando comparado a qualidade alta. Além disso, apesar de haver um aumento para o uso da TEO, esse aumento não foi tão considerável quanto o do janelamento, apresentando apenas uma média de acurácias de 65,27%.

No último teste do k-NN, foram utilizadas exemplos com as três qualidades. Foi obtido o esperado, houve a diminuição na capacidade de generalização do modelo, com apenas 65,16% e 56,03% de acurácia para o janelamento e TEO respectivamente. Contudo, foi obtido uma EER razoável de 0,32 com o janelamento.

**Tabela 4.1** – Resultados obtidos pelo classificador k-NN, com  $k = 10$ , para cada tipo de conjunto de treino e teste. QP para qualidade de *replay* perfeita, QA para qualidade de *replay* alta, QB para qualidade de *replay* baixa e Todas foi utilizados um conjunto de exemplos com as três qualidades.

	Janela	TEO
Acurácia media - QP (%)	54,55	50,06
Acurácia media - QA (%)	63,92	59,72
Acurácia media - QB (%)	87,76	65,27
Acurácia media - Todas (%)	65,34	56,03
EER - QP	0,52	0,44
EER - QA	0,26	0,33
EER - QB	0,12	0,32
EER - Todas	0,32	0,48

### 4.3 Testes SVM

Foram utilizados os mesmo princípios dos teste do k-NN para o classificador SVM. No primeiro teste, percebe-se um aumento pequeno na acurácia, 2,59% para o janelamento e 5,6% para a TEO, enquanto a EER se manteve bem parecida.

Para o segundo teste, o classificador SVM mostrou-se uma superioridade quando a uma queda da

qualidade do dispositivo de ataque, tendo uma EER de 0,08 e acurácia média de 87,37% para o janelamento, e, para a característica TEO, uma acurácia de 64,39% e EER de 0,28. Com isso, há alguns indícios que a TEO não é tão afetada pela qualidade do dispositivo quanto o janelamento.

Para o terceiro teste, quando diminuimos a qualidade para baixa, o modelo SVM ainda se mostrou o mais promissor, com uma acurácia média de 97,21% e teve uma EER de 0.0 para o janelamento, a qual nós mostra que em determinado conjunto de treino e validação houve uma taxa de 100% de acertos. Além disso, houve um aumento na acurácia quando utilizamos a TEO em gravações de qualidade baixa, obtendo uma acurácia média de 71,55%. Entretanto, a TEO não se mostrou superior ao janelamento até o momento.

No último teste, houve a mesma decaída na classificação que ocorreu no k-NN. Entretanto, o modelo SVM foi menos afetado quando utilizada todas as qualidades, para o janelamento, foi obtido uma acurácia média de 71,23%, superior ao classificador k-NN mesmo quando esse utilizou apenas ataques com dispositivos de qualidade alta, o que evidencia uma melhor eficiência desse classificador.

**Tabela 4.2** – Resultados obtidos pelo classificador SVM, para cada tipo de conjunto de treino e teste. QP para qualidade de *replay* perfeita, QA para qualidade de *replay* alta, QB para qualidade de *replay* baixa e Todas foi utilizados um conjunto de exemplos com as três qualidades.

	Janela	TEO
Acurácia média - QP (%)	57,14	55,66
Acurácia média - QA (%)	87,37	64,39
Acurácia média - QC (%)	97,21	71,55
Acurácia média - Todas (%)	71,23	60,38
EER - QP	0,48	0,44
EER - QA	0,08	0,28
EER - QB	0,0	0,28
EER - Todas	0,24	0,28

## Capítulo 5

### Conclusões

O objetivo de construir um modelo para a detecção de ataque via *replays* foi realizado. Além disso, o estudo desse trabalho mostrou uma possível vantagem do classificador SVM sobre o k-NN, quando trabalhada no processamento de voz em relação a detecção de *replays*.

Ao analisar as acurácias obtidas por ambos modelos, é notável uma maior eficiência no modelo SVM, em qualquer situação, a qual é bem mais agravável ao compararmos os testes realizados com a qualidade alta, onde o janelamento teve uma diferença de 23,45% com o modelo k-NN. Com isso, o classificador SVM teve maior eficiência para detectar os ataques via *replays*, mostrando-se um classificador com boas qualidades para trabalhar em sistemas de contra-medida.

Para as características, a TEO não se mostrou uma *feature* eficiente para ambos os classificadores, mesmo quando diminuído a qualidade do *replay*. Entretanto, o janelamento se mostrou uma boa característica quando as qualidades são alta e baixa, e, quando é utilizado diversos ataques com diversas qualidades, o janelamento em conjunto a SVM conseguiu generalizar razoavelmente.

Em relação a outros trabalhos, o uso de características do cepstrum, como visto em [1], [3] e [4] mostrou maior eficiência para a detecção de ataques quando comparadas aos modelos utilizados nesse trabalho. Entretanto, o resultado do classificador SVM junto com o janelamento quando utilizadas todas as qualidades, foi satisfatório, obtendo uma acurácia razoável de 71,23% e uma EER de 0,24 utilizando características simples.

## Referências

- [1] WITKOWSKI, M. et al. Audio Replay Attack Detection Using High-Frequency Features. Interspeech 2017. **Anais...** In: INTERSPEECH 2017. ISCA, 20 ago. 2017.
- [2] HIMAWAN, I. et al. Deep domain adaptation for anti-spoofing in speaker verification systems. **Computer Speech & Language**, v. 58, p. 377–402, nov. 2019.
- [3] TAK, H.; PATIL, H. Novel Linear Frequency Residual Cepstral Features for Replay Attack Detection. Interspeech 2018. **Anais...** In: INTERSPEECH 2018. ISCA, 2 set. 2018.
- [4] KAMBLE, M. R.; PATIL, H. A. Detection of replay spoof speech using teager energy feature cues. **Computer Speech & Language**, v. 65, p. 101140, jan. 2021.
- [5] KAMBLE, M. R.; PATIL, H. A. Analysis of Reverberation via Teager Energy Features for Replay Spoof Speech Detection. ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). **Anais...** In: ICASSP 2019 - 2019 IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING (ICASSP). Brighton, United Kingdom: IEEE, maio 2019.
- [6] JAVED, A. et al. Towards protecting cyber-physical and IoT systems from single- and multi-order voice spoofing attacks. **Applied Acoustics**, v. 183, p. 108283, 1 dez. 2021.
- [7] JAIN, A. K.; ROSS, A.; PANKANTI, S. Biometrics: A Tool for Information Security. **IEEE Transactions on Information Forensics and Security**, v. 1, n. 2, p. 125–143, jun. 2006.
- [8] Nuance Vocal Password. Disponível em: <<https://www.nuance.com/en-gb/omni-channel-customer-engagement/security/multi-modal-biometrics/vocalpassword.html>>. Acesso em: 22 maio 2021.
- [9] WU, Z. et al. Spoofing and countermeasures for speaker verification: A survey. **Speech Communication**, v. 66, p. 130–153, fev. 2015.

- [10] Automatic Speaker Verification Spoofing And Countermeasures Challenge. Disponível em: <<https://www.asvspoof.org/>> . Acesso em: 24 maio 2021.
- [11] YAMAGISHI, J. et al. ASVspoof 2019: The 3rd Automatic Speaker Verification Spoofing and Countermeasures Challenge database. 17 jun. 2019.
- [12] AUDACITY SOFTWARE: Editor de áudio. Versão 2.4.1. [S. l.]: Audacity Team, 28 maio 2000. Disponível em: <https://www.audacityteam.org/download/>. Acesso em: 11 jul. 2020.
- [13] FRE:AC: Conversor de áudio gratuito. Versão 1.1.2a. [S. l.]: Robert Kausch, 21 maio 2001. Disponível em: <https://www.freac.org/downloads-mainmenu-33>. Acesso em: 6 maio 2020.
- [14] DEVASAHAYAM, Suresh R. **Signals and systems in biomedical engineering: signal processing and physiological systems modeling**. Springer Science & Business Media, 2000.
- [15] GUIDO, R. C. Enhancing teager energy operator based on a novel and appealing concept: Signal mass. **Journal of the Franklin Institute**, v. 356, n. 4, p. 2346–2352, 1 mar. 2019.
- [16] KAISER, J. F. On a simple algorithm to calculate the “energy” of a signal. International Conference on Acoustics, Speech, and Signal Processing. **Anais...** In: INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING. abr. 1990.
- [17] TODISCO, M. et al. ASVspoof 2019: Future Horizons in Spoofed and Fake Audio Detection. **arXiv:1904.05441 [cs, eess]**, 14 abr. 2019.
- [18] KINNUNEN, T. et al. t-DCF: a Detection Cost Function for the Tandem Assessment of Spoofing Countermeasures and Automatic Speaker Verification. **arXiv:1804.09618 [cs, eess, stat]**, 11 abr. 2019.
- [19] Validação Cruzada. Disponível em: <[https://docs.aws.amazon.com/pt\\_br/machine-learning/latest/dg/cross-validation.html](https://docs.aws.amazon.com/pt_br/machine-learning/latest/dg/cross-validation.html)>. Acesso em: 22 jun. 2021.
- [20] KOHAVI, Ron et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: **Ijcai**. jan. 1995. p. 1137-1145.
- [21] WAVE and AVI Codec Registries. Disponível em: <<https://datatracker.ietf.org/doc/html/rfc2361>>. Acesso em: 20 jun. 2021.

- [22] IBM CORPORATION AND MICROSOFT CORPORATION. Multimedia Programming Interface and Data Specifications 1.0. 1991.
- [23] KIANG, M. Y. A comparative assessment of classification methods. **Decision Support Systems**, v. 35, n. 4, p. 441–454, jul. 2003.
- [24] THARWAT, A. Classification assessment methods. **Applied Computing and Informatics**, v. 17, n. 1, p. 168–192, 4 jan. 2021.
- [25] KUHN, M.; JOHNSON, K. **Applied predictive modeling**. New York: Springer, 2013.
- [26] TEKNOMO, K. K-Nearest Neighbors Tutorial. Disponível em: <<https://people.revoledu.com/kardi/tutorial/KNN/>>. Acesso em: 28 mai. 2021
- [27] PRASATH, V. B. et al. Distance and Similarity Measures Effect on the Performance of K-Nearest Neighbor Classifier–A Review. **arXiv preprint arXiv:1708.04321**, 2017.
- [28] FIX, E.; HODGES, J.L. Discriminatory analysis. Nonparametric discrimination consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, TX, USA, 1951.
- [29] PRASATH, V. B. et al. Mahmoud Bashir Alhasanat, and Hamzeh S. Eyal Salman. “Distance and Similarity Measures Effect on the Performance of K-Nearest Neighbor Classifier–A Review.”. **Big Data**, 2019.
- [30] JAMES, G. et al. (EDS.). **An introduction to statistical learning: with applications in R**. New York: Springer, 2013.