UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

MARCOS CORDEIRO JÚNIOR

DETECÇÃO DE EDIÇÕES EM ÁUDIOS BASEADA NA ANÁLISE TEMPO-FREQUÊNCIA E EM REDES NEURAIS CONVOLUCIONAIS

DISSERTAÇÃO

CURITIBA

2023

MARCOS CORDEIRO JÚNIOR

DETECÇÃO DE EDIÇÕES EM ÁUDIOS BASEADA NA ANÁLISE TEMPO-FREQUÊNCIA E EM REDES NEURAIS CONVOLUCIONAIS

Audio tampering detection based on time-frequency analysis and convolutional neural networks

Dissertação apresentada como requisito para obtenção do grau de Mestre em Engenharia Elétrica e Informática Industrial, do Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial, da Universidade Tecnológica Federal do Paraná (UTFPR).

Orientador: Prof. Dr. Daniel Rodrigues Pipa

CURITIBA

2023

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Curitiba

MARCOS CORDEIRO JUNIOR

DETECÇÃO DE EDIÇÕES EM ÁUDIOS BASEADA NA ANÁLISE TEMPO-FREQUÊNCIA E EM REDES NEURAIS CONVOLUCIONAIS.

Trabalho de pesquisa de mestrado apresentado como requisito para obtenção do título de Mestre Em Ciências da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Engenharia De Computação.

Data de aprovação: 28 de Novembro de 2023

Dr. Daniel Rodrigues Pipa, Doutorado - Universidade Tecnológica Federal do Paraná

Dr. Andre Eugenio Lazzaretti, Doutorado - Universidade Tecnológica Federal do Paraná

Dr. Luiz Wagner Pereira Biscainho, Doutorado - Universidade Federal do Rio de Janeiro (Ufrj)

Documento gerado pelo Sistema Acadêmico da UTFPR a partir dos dados da Ata de Defesa em 28/11/2023.

RESUMO

CORDEIRO JR, Marcos. **Detecção de edições em áudios baseada na análise tempo-frequência e em redes neurais convolucionais**. 2023. 63 f. Dissertação (Mestrado em Engenharia Elétrica e Informática Industrial) – Universidade Tecnológica Federal do Paraná. Curitiba, 2023.

A detecção de edições é um dos mais importantes tópicos na análise forense de áudios digitais. A interpolação (*splicing*), caracterizada pela inserção de um trecho de sinal proveniente de um áudio distinto no registro de áudio original, é incluída entre as categorias mais recorrentes de adulterações. As redes neurais convolucionais (CNNs) têm demonstrado eficácia em diversas tarefas de processamento de áudio, o que motiva a pesquisa por diferentes formas de obtenção dos dados de entrada. O espectrograma é uma representação útil para a visualização da evolução temporal do espectro de frequências de um áudio, sendo que diferentes técnicas de processamento de sinais podem ser utilizadas para a sua obtenção. No presente trabalho, foi realizado o desenvolvimento de um modelo de detecção automática de interpolação em áudios digitais com o uso de CNNs. O espectrograma dos áudios, calculado através de diferentes técnicas: transformada de Fourier de tempo curto (STFT) na escala linear, STFT na escala mel e transformada Q constante (CQT), foi diretamente fornecido à rede como dado de entrada. Um estudo comparativo foi conduzido avaliando o impacto da escolha da representação no domínio tempo-frequência no desempenho do modelo em classificar corretamente os áudios originais e editados.

Palavras-chave: Detecção de edição em áudios. Redes neurais convolucionais. Análise tempo-frequência.

ABSTRACT

CORDEIRO JR, Marcos. **Audio tampering detection based on time-frequency analysis and convolutional neural networks**. 2023. 63 p. Dissertation (Master's Degree in Graduate Program in Electrical Engineering and Industrial Informatics) – Universidade Tecnológica Federal do Paraná. Curitiba, 2023.

Tampering detection is one of the most important topics in forensic analysis of digital audio. Splicing corresponds to the insertion of a segment of signal from a different audio into the original audio record and is included among the most common categories of tampering. Convolutional neural networks (CNNs) have demonstrated effectiveness in various audio processing tasks, which motivates research into different methods of obtaining input data. The spectrogram is a useful representation for visualizing the temporal evolution of the frequency spectrum of an audio, with different signal processing techniques available for its generation. In this study, the development of an automatic splicing detection model in digital audio using CNNs was carried out. The audio spectrogram, computed using different techniques such as Short-Time Fourier Transform (STFT) on a linear scale, STFT on a mel scale, and Constant Q Transform (CQT), was directly provided to the network as input data. A comparative study was conducted to evaluate the impact of the choice of time-frequency representation on the model's performance in correctly classifying the original and edited audios.

Keywords: Audio tampering detection. Convolutional Neural Networks. Time-frequency analysis.

LISTA DE ALGORITMOS

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

ABREVIATURAS

Acc	Accuracy
Cap.	Capítulo
Fig.	Figura

SIGLAS

AEM	Absolute Error Map
AI	Artificial Intelligence
CQT	Constant Q Transform
CNN	Convolutional Neural Network
DL	Deep Learning
DFT	Discrete Fourier Transform
DTFT	Discrete-Time Fourier Transform
ENF	Electrical Network Frequency
FFT	Fast Fourier Transform
FP	False Positive
FN	False Negative
FT	Fourier Transform
LFCC	Linear Frequency Cepstral Coefficients
LJ	LJSpeech Dataset
MFCC	Mel-Frequency Cepstral Coefficients
NLP	Natural Language Processing
RIR	Room Impulse Response
RMS	Root Mean Square
SC	Speech Commands Dataset
SVD	Singular Value Decomposition
SNR	Signal-to-Noise Ratio
STFT	Short-Time Fourier Transform
TP	True Positive
TN	True Negative

ACRÔNIMOS

AUC	Area Under the Curve
ReLU	Rectified Linear Unit
ROC	Receiver Operating Characteristic

SUMÁRIO

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

Áudios digitais representam, atualmente, uma das principais formas de transmissão de conteúdo e, como consequência, se tornaram uma importante fonte de evidência em procedimentos judiciais. Com a grande disponibilidade de *softwares* de edição, os arquivos de áudio podem ser facilmente manipulados por usuários amadores sem que traços da adulteração possam ser facilmente observados.

A análise forense em áudios envolve a interpretação científica dos registros de áudios obtidos a partir de investigações civis formais ou processos legais criminais (MAHER, 2010). Nesse contexto, o exame de verificação de edição é um processo que visa encontrar elementos indicativos de adulterações que possam modificar o sentido do conteúdo original de um determinado áudio.

Entre as categorias de edições existentes, o processo de interpolação (*splicing*) é comumente verificado. A interpolação corresponde à inserção de um trecho de sinal oriundo de um áudio distinto no registro de áudio original.

Os métodos de verificações de edições podem ser categorizados entre os que são baseados na análise do *container* e os que são baseados na análise do conteúdo (ZAKARIAH *et al.*, 2018). Os métodos baseados no *container* avaliam as informações referentes à estrutura do arquivo e dos metadados. Essa abordagem, apesar de importante para a autenticação, nem sempre é passível de ser realizada, pois depende de informações que nem sempre estão disponíveis e podem ser alteradas/removidas. De outro modo, os métodos baseados no conteúdo focam exclusivamente na análise do sinal e das propriedades do áudio, consistindo em um tema recorrente de pesquisas na área de ciências forenses.

Ao longo dos últimos anos, diversos métodos foram propostos para a detecção de adulterações em registros de áudio baseada no conteúdo, incluindo: verificação da continuidade de fase do sinal da rede elétrica (*Electrical Network Frequency* - ENF) (HUA *et al.*, 2016), análise da variação espectral do padrão do ruido de fundo (PAN *et al.*, 2012), estudo da correlação estatística nas dependências lineares dos pontos do sinal através da decomposição em valores singulares (*Singular Value Decomposition* - SVD) (SHI; MA, 2011), análise de inconsistências no tempo de reverberação em uma gravação de áudio (CAPOFERRI *et al.*, 2020). Esses métodos exigem,

como etapa de pré-processamento, a extração manual do vetor de características (*features*).

Entre as tendências observadas nas pesquisas relacionadas ao tema, a aprendizagem profunda (*Deep Learning* - DL) desponta como uma área da inteligência artificial potencialmente capaz de superar as técnicas anteriormente desenvolvidas. As redes neurais convolucionais (*Convolutional Neural Networks* - CNNs) possuem a capacidade de extração automática das *features* e foram utilizadas em conjunto com processamento do espectrograma da transformada de Fourier de tempo curto (*Short Time Fourier Transform* - STFT) para a detecção de interpolações (JADHAV *et al.*, 2019) e réplicas (USTUBIOGLU *et al.*, 2022) em áudios.

O espectrograma da STFT, tanto na escala linear como na escala mel, é uma representação no domínio tempo-frequência particularmente útil devido à natureza não estacionária do sinal sonoro. No entanto, uma representação alternativa derivada da análise tempo-frequência, o espectrograma da transformada Q constante (*Constant Q Transform* - CQT), foi aplicada com sucesso em diferentes tarefas de processamento de sinais de áudio, como classificação de cenas acústicas (LIDY; SCHINDLER, 2016), classificação de eventos de áudio (MCLOUGHLIN *et al.*, 2020) e detecção de *deepfakes* (ZIABARY; VEISI, 2021).

À medida que as redes neurais convolucionais se apresentam como uma ferramenta poderosa para a análise de áudio, a pesquisa por métodos diferentes de pré-processamento dos dados de entrada torna-se imperativa para otimizar o desempenho da rede diante da diversidade de características presentes nos sinais auditivos. Assim, esta dissertação busca não apenas explorar a combinação da CQT e da CNN para a detecção de interpolações em áudios, mas também efetuar comparações com as representações convencionais no domínio tempo-frequência.

1.2 OBJETIVOS

1.2.1 Objetivo geral

O presente trabalho propõe o desenvolvimento e estudo comparativo de diferentes métodos para a detecção automática de interpolação em áudios digitais com a utilização de redes neurais convolucionais. Serão analisadas três abordagens para a obtenção da representação no domínio tempo-frequência: a transformada de fourier de tempo curto (STFT) na escala linear, STFT na escala mel e a transformada Q constante (CQT). O desempenho do modelo será avaliado em um conjunto de áudios originais e editados a partir dos *datasets* LJSpeech (ITO; JOHNSON, 2017) e SpeechCommands (WARDEN, 2018).

1.2.2 Objetivos específicos

- Efetuar o pré-processamento dos áudios constantes nos *datasets* utilizados.
- Gerar uma base de dados contendo áudios originais e editados.
- Construir os espectrogramas dos áudios através de três diferentes representações: STFT em escala linear, STFT em escala mel e CQT.
- Extrair as *features* dos espectrogramas com redes neurais convolucionais.
- Efetuar a tarefa de classificação entre áudios originais e editados.
- Comparar o resultado de detecção de edição entre as três diferentes representações anteriormente citadas.
- Testar a robustez do modelo através da aplicações de diferentes operações de pós-processamento nos áudios.

1.3 ESTRUTURA DO TRABALHO

Após esta introdução, o Capítulo 2 aborda os fundamentos teóricos relevantes para o estudo em questão. Em seguida, o Capítulo 3 apresenta alguns do trabalhos relacionados ao tema. O Capítulo 4 descreve a metodologia e os recursos empregados na elaboração deste trabalho. No Capítulo 5, são apresentados e discutidos os principais resultados obtidos. Por fim, o Capítulo 6 abrange as conclusões e as sugestões para trabalhos futuros.

2 FUNDAMENTOS TEÓRICOS

2.1 CIÊNCIAS FORENSES

Devido à natureza multidisciplinar da área e à diversidade dos sistemas legais, a definição de ciências forenses, ou criminalística, pode ser objeto de debate e controvérsia por diferentes profissionais e acadêmicos. Na definição de Houck e Siegel (2009), o termo ciência forense engloba o estudo científico da associação entre pessoas, lugares e objetos envolvidos em atividades criminosas, e essas disciplinas científicas desempenham um papel fundamental na investigação e no julgamento de casos criminais e civis.

O uso moderno do termo "forense"como adjetivo faz referência a qualquer atividade que possa ser levada a um tribunal, enquanto a ciência pode ser definida como um método de estudo utilizado na tentativa de descrever o universo físico baseado na identificação e no uso de padrões repetidos para o estabelecimento de regrais gerais (INMAN; RUDIN, 2000).

As ciências forenses abrangem uma ampla gama de disciplinas e áreas de especialização, tais como: antropologia, balística, computação, documentoscopia, engenharias, genética e toxicologia.

2.1.1 Análise forense de áudios

A área de análise forense de áudios envolve a aquisição, análise e avaliação de gravações de áudio utilizadas como evidência em tribunais ou para investigações, sendo usualmente empregada para a determinação da autenticidade e verificação da integridade das provas apresentadas em casos civis ou criminais (ZAKARIAH *et al.*, 2018).

Não obstante ser um campo especializado, inclui diversas subáreas que abarcam conhecimentos de diferentes disciplinas, incluindo: aprimoramento de áudio, análise de voz e autenticação de áudio ou verificação de edição (ZJALIC, 2020).

O aprimoramento de áudio envolve um conjunto de técnicas aplicadas a fim de melhorar a qualidade e a inteligibilidade do registro. Ajustes de amplitude e velocidade de reprodução, remoções de ruídos de fundo e aplicações de filtros são algumas das operações que podem ser utilizadas para tal finalidade.

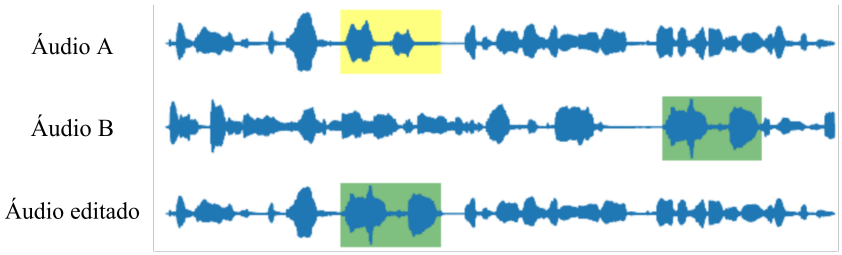
A análise de voz abrange o exame de comparação entre vozes contidas em diálogos

armazenados em arquivos de áudio para atribuir um grau de compatibilidade aos locutores. Esse exame envolve conceitos de acústica, fonética e linguística.

O exame de verificação de edição é um processo que visa encontrar elementos indicativos de adulterações ou manipulações que possam modificar o sentido do conteúdo original de um determinado arquivo de áudio. Para a determinação da originalidade de um registro questionado, são realizadas análises da estrutura do arquivo e metadados, além de análises das propriedades do sinal de áudio (ZAKARIAH *et al.*, 2018).

Um registro de áudio pode ser submetido a diversos processos de edições, tais como interpolações (*splicing*), réplicas (*copy-move*), supressões e operações de pós-processamento. O processo de interpolação, ilustrado na Figura 1, é um dos tipos mais comuns de edição e corresponde à inserção de um trecho de sinal oriundo de um áudio distinto no registro de áudio original.

Figura 1 – Exemplo de interpolação de um sinal de áudio.



Fonte: Autoria própria.

2.2 PROCESSAMENTO DE SINAIS

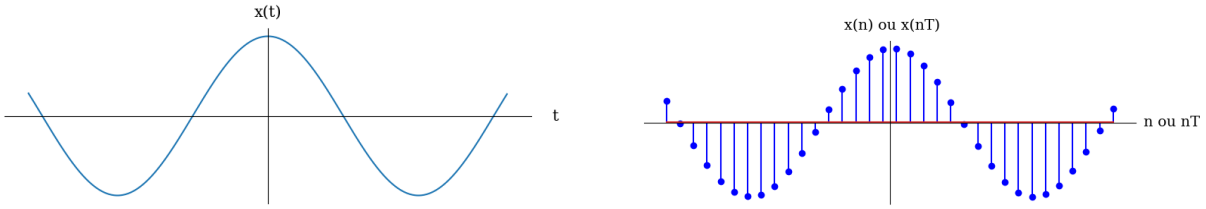
2.2.1 Análise de sinais no domínio do tempo

A análise de sinais no domínio do tempo permite o estudo das características contidas em um sinal de áudio ao longo do eixo temporal, com o objetivo de extrair informações sobre o conteúdo da forma de onda sonora.

Um sinal $x(t)$ é um sinal de tempo contínuo se ele for definido para cada instante de tempo. Um sinal em tempo discreto é uma sequência de valores que é definida apenas em pontos discretos no tempo e, quando obtido através da amostragem de um sinal $x(t)$ contínuo no tempo, pode ser definido por $x(n)$ ou $x(nT)$, onde T é o período (intervalo) de amostragem e n representa a variável discreta inteira. A Figura 2 apresenta um exemplo da representação gráfica

dos sinais.

Figura 2 – Representação de um sinal em tempo continuo e em tempo discreto.



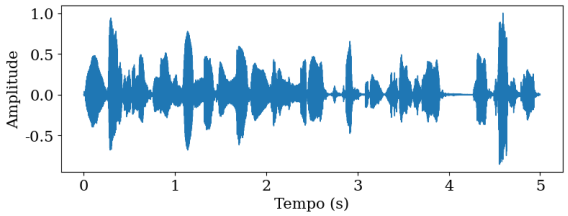
(a) Sinal em tempo contínuo. **(b) Sinal em tempo discreto.**

Fonte: Autoria própria.

Um som é gerado por meio de um objeto vibrante que causa deslocamentos e oscilações das moléculas de ar. A pressão alternante se propaga pelo ar como uma onda, desde sua origem até um ouvinte ou microfone (MÜLLER, 2015). Graficamente, a forma de onda de um som representa o desvio da pressão sonora em relação a uma linha de base, que corresponde à média das variações de pressão ao longo do tempo. O termo áudio é utilizado para a descrição de sons captados pela audição humana.

Um sinal digital de áudio é um exemplo de sinal em tempo discreto, correspondendo a uma representação numérica digital de um som. Ele é composto por uma sequência de amostras discretas que capturam o desvio da pressão sonora. Cada amostra representa a amplitude ou intensidade do som em um determinado instante de tempo. A Figura 3 ilustra a forma de onda de um sinal de voz.

Figura 3 – Forma de onda de um áudio.



Fonte: Autoria própria.

O áudio digital é obtido por meio de um processo de conversão analógico-digital (A/D), em que um sinal sonoro contínuo é amostrado e quantizado para representação digital. O processo de amostragem corresponde à discretização no tempo e o processo de quantização corresponde à discretização da amplitude (MÜLLER, 2015). O sinal acústico, composto de ondas sonoras, é transformado em sinal elétrico (analógico) por um transdutor (microfone) e posteriormente

convertido em um sinal digital. O sinal resultante pode então ser armazenado, processado e transmitido por dispositivos eletrônicos.

2.2.2 Análise de sinais no domínio da frequência

Embora seja possível obter informações úteis sobre um sinal de áudio ao analisá-lo no domínio do tempo (forma de onda), a análise no domínio da frequência oferece uma perspectiva abrangente e detalhada. Através do exame do espectro do sinal, diversas características e padrões sonoros podem ser melhor visualizados. Além disso, muitas técnicas de processamento de sinais, como filtragem, equalização e compressão de áudio, são mais eficientes e convenientes de serem aplicadas no domínio da frequência.

A série de Fourier é uma técnica matemática que permite representar um sinal periódico como uma superposição ponderada de componentes senoidais de diferentes frequências, cada uma com sua própria amplitude e fase (HAYKIN; VEEN, 2001). Essas componentes senoidais, chamadas de harmônicos, são múltiplos inteiros da frequência fundamental do sinal periódico.

A série de Fourier, descrita em (1), pode ser utilizada para representar um sinal periódico $x(t)$ através da seguinte fórmula compacta (LATHI, 2006)

$$x(t) = \sum_{n=-\infty}^{\infty} c_n \cdot e^{jn\omega_0 t} \tag{1}$$

com

$$c_n = \frac{1}{T_0} \int_{T_0} x(t) \cdot e^{-jn\omega_0 t}\, dt \tag{2}$$

onde ω_0 é a frequência fundamental angular (rad/s) e T_0 é o período fundamental (s). A relação entre a frequência e o período é dada por: $\omega_0 = \frac{2\pi}{T_0}$.

Os sinais de áudio são caracterizados por serem não periódicos. Enquanto a série de Fourier é limitada para representar sinais periódicos, a transformada de Fourier permite capturar a natureza dinâmica dos sinais de áudio no domínio da frequência.

A transformada de Fourier (FT) de um sinal $x(t)$ contínuo no tempo pode ser expressa por (LATHI, 2006)

$$X_{FT}(\omega) = \int_{-\infty}^{\infty} x(t) \cdot e^{-j\omega t} dt \tag{3}$$

onde ω é a variável de frequência angular, medida em radianos/segundo (rad/s).

A integral de Fourier descrita em (3) pode ser compreendida como um caso particular da série de Fourier para um sinal com frequência fundamental tendendo a zero ou período fundamental tendendo ao infinito.

O Teorema da Amostragem estabelece uma condição fundamental para a amostragem de sinais contínuos no domínio do tempo. De acordo com o teorema, para que um sinal contínuo seja perfeitamente reconstruído a partir de suas amostras, a taxa de amostragem deve ser ao menos duas vezes maior do que a maior frequência presente no sinal.

Dessa forma, se um sinal $x(t)$ contínuo no tempo tem largura de banda limitada em B Hz, ou seja, a sua transformada de Fourier $X_{FT}(\omega) = 0$ para $|\omega| > 2\pi B$, então a reconstrução de $x(t)$ pode ser realizada a partir do sinal em tempo discreto $x(n)$ se a frequência de amostragem $f_s > 2B$ (LATHI, 2006).

A transformada de Fourier de tempo discreto (DTFT) aplicada ao sinal $x(n)$ pode ser expressa por (LATHI, 2006)

$$X_{DTFT}(\Omega) = \sum_{n=-\infty}^{\infty} x(n) \cdot e^{-j\Omega n} \tag{4}$$

onde Ω é a frequência angular, medida em radianos/amostra (rad/amostra) e n é a variável discreta inteira que corresponde ao índice do tempo.

Ao analisar a equação (4), verifica-se que a somatória contém um número infinito de termos e que, apesar de a variável de tempo estar no domínio discreto, a variável de frequência angular Ω é contínua. Para o processamento de sinais digitais de áudio, considera-se que o sinal envolve apenas um número finito N de termos iniciando em $n = 0$. Ainda, também é necessária a amostragem da variável de frequência.

Assim, a transformada discreta de Fourier (DFT) pode ser representada por (5) (DINIZ *et al.*, 2014)

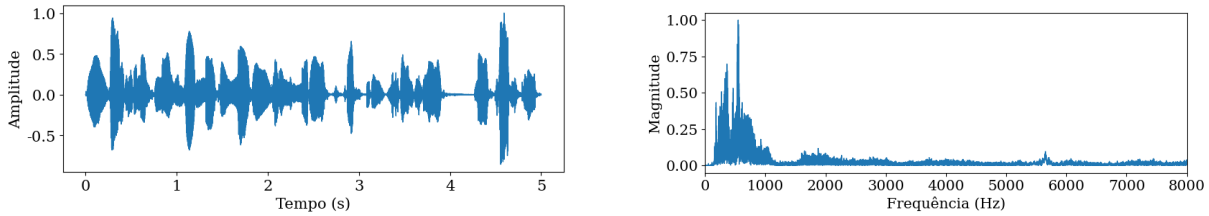
$$X_{DFT}(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-j\frac{2\pi kn}{N}} \tag{5}$$

onde N é o número de amostras do sinal e k é a variável discreta inteira que corresponde ao índice da frequência.

Considerando que a função $X_{DFT}(k)$ é complexa, podem ser gerados os espectros de amplitude e de fase. Destarte, o espectro de amplitude de Fourier de um sinal digital pode ser obtido através de uma representação do valor absoluto $|X_{DFT}(k)|$ da função complexa em função das frequências. Em outras palavras, a amplitude representa a magnitude das componentes de frequência do sinal.

A Figura 4 apresenta a forma de onda de um sinal de áudio e o espectro de amplitude normalizada de frequências correspondente, obtido através da aplicação da transformada de Fourier.

Figura 4 – Sinal de áudio e respectivo espectro de amplitude de frequências.



(a) Forma de onda. **(b) Espectro de frequências.**

Fonte: Autoria própria.

Um algoritmo denominado transformada rápida de Fourier (FFT) foi desenvolvido por Cooley e Tukey (1965) com o objetivo de diminuir a complexidade do cálculo da DFT. O algoritmo, amplamente utilizado no processamento digital de sinais, reduz o número de cálculos da ordem de N^2 para a ordem de $N \log_2 N$.

2.2.3 Análise de sinais no domínio tempo-frequência

Conforme exposto na subseção anterior, a transformada de Fourier revela as informações sobre as componentes de frequência presentes em um sinal ao longo de sua totalidade. Entretanto, o espectro resultante não contém informação temporal, o que significa que é possível identificar quais componentes de frequência estão presentes no sinal, mas não é possível determinar em quais momentos ocorrem suas transições.

O espectrograma é um gráfico que mede a densidade espectral de energia, consistindo em uma ferramenta útil para a visualização da evolução temporal do espectro de frequências do sinal. A construção do espectrograma é realizada através da transformada de Fourier de tempo curto (STFT), que consiste na segmentação do sinal em trechos menores e do cálculo sucessivo da DFT (ou da FFT) para uma janela $w(n)$ que se desloca no tempo. A expressão para o cálculo da STFT com a DFT pode ser definida como (MÜLLER, 2015)

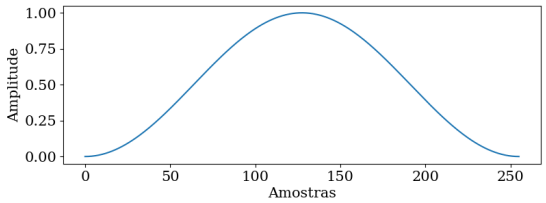
$$X_{STFT}(m,k) = \sum_{n=0}^{N-1} x(n + mH) \cdot w(n) \cdot e^{\frac{-j2\pi kn}{N}} \tag{6}$$

onde m é a variável discreta que corresponde ao índice de deslocamento da janela e H é o tamanho do passo de deslocamento entre janelas consecutivas.

Os eixos horizontais e verticais do espectrograma são representados, respectivamente, pelo tempo e frequência. O valor da magnitude em uma determinada frequência e um determinado tempo é equivalente ao valor quadrático do módulo $|X_{STFT}(m,k)|^2$ da função complexa, e é representado através de uma escala logarítmica previamente definida de cores.

Existem diferentes funções de janelamento que podem ser aplicadas ao sinal para o cálculo da STFT, como a janela quadrangular, por exemplo. A janela de Hann, ilustrada na Figura 5, é uma função simétrica de amplitude decrescente em direção às bordas, definida por: $w(n) = 0{,}5 \cdot \left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right)$. O propósito da sua utilização é a diminuição do vazamento do conteúdo espectral.

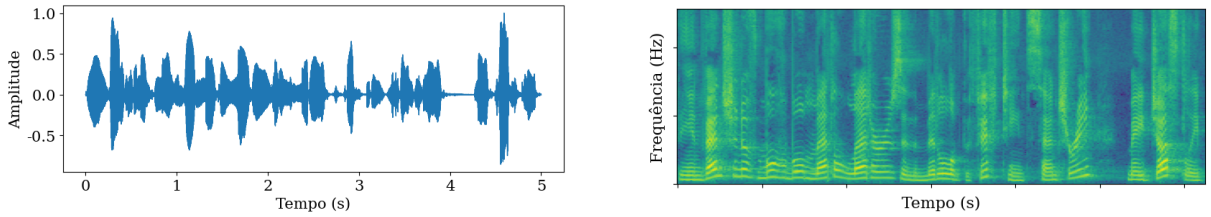
Figura 5 – Janela de Hann.



Fonte: Autoria própria.

A escolha do tamanho da janela com um maior número de pontos implica em uma maior resolução em frequência e em uma menor resolução temporal. De modo oposto, uma janela menor resulta em maior resolução temporal e menor resolução em frequência. A Figura 6(b) apresenta o espectrograma de um sinal, gerado com janelamento de Hann, $N = 512$ e $H = N/2$. O valores de magnitude foram representados em uma escala normalizada.

Figura 6 – Forma de onda do sinal e respectivo espectrograma da STFT.



(a) Forma de onda. **(b) Espectrograma da STFT.**

Fonte: Autoria própria.

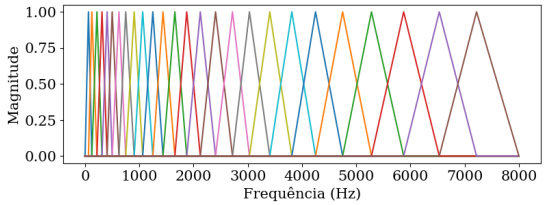
No espectrograma, apesar de a magnitude ser usualmente apresentada em escala logarítmica, a escala de frequências é disposta de forma linear. A escala mel (derivada da palavra *melody*) foi desenvolvida experimentalmente para melhor representar como diferentes frequências eram percebidas pelo aparelho auditivo humano. A conversão de Hertz para mel é realizada através de

$$m = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{7}$$

O mel-espectrograma é gerado através da aplicação de um banco de filtros triangulares na STFT, com frequências centrais definidas com base na escala mel. A Figura 7 apresenta o

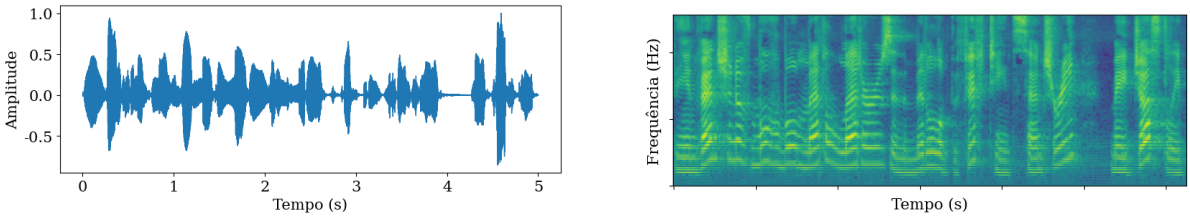
exemplo de um banco de 26 filtros para um sinal amostrado em 16 KHz. Um mel-espectrograma, obtido do espectrograma anterior com a aplicação de 257 filtros, é visualizado na Figura 8(b).

Figura 7 – Banco de filtros mel.



Fonte: Autoria própria.

Figura 8 – Forma de onda do sinal e respectivo mel-espectrograma da STFT.



(a) Forma de onda. **(b) Mel-Espectrograma da STFT.**

Fonte: Autoria própria.

A transformada Q constante, proposta inicialmente por Brown (1991) para representação de sinais musicais, corresponde a uma técnica de análise onde, diferentemente da STFT, a resolução é variável e as frequências são espaçadas de maneira logarítmica através da variação do comprimento da janela de análise. Essa representação não linear se aproxima melhor da percepção auditiva humana.

As frequências centrais são calculadas por: $f_k = (2^{1/b})^k f_{min}$, onde o valor de b é equivalente ao número de filtros dentro de cada oitava na escala musical e f_{min} denota a menor frequência analisada. A constante Q fixa a proporção entre as frequências centrais adjacentes

$$Q = \frac{f_k}{f_{k+1} - f_k} = \frac{f_k}{\Delta f_k} = (2^{1/b} - 1)^{-1} \tag{8}$$

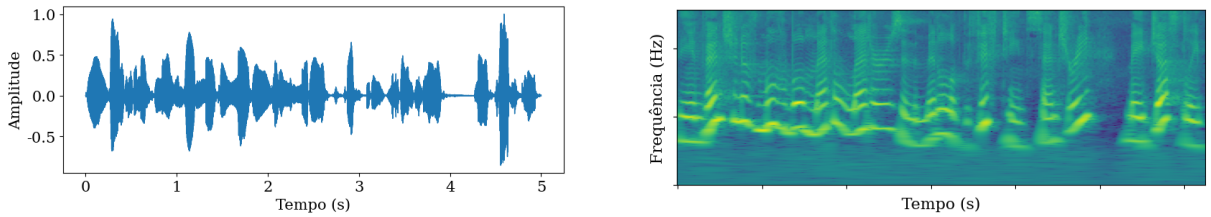
A variação do comprimento da janela de análise é dada pela relação: $N_k = Qf_s/f_k$, onde f_s é a frequência de amostragem. Desta forma, a transformada é definida por (BROWN, 1991)

$$X_{CQT}(m,k) = \frac{1}{N_k} \sum_{n=0}^{N_k-1} x(n) \cdot w(k,m) \cdot e^{\frac{-j2\pi Qn}{N_k}} \tag{9}$$

onde N_k é tamanho da janela de análise no índice de frequência k e $w(k,m)$ é a janela de análise aplicada no índice k e no índice m.

Um exemplo do espectrograma resultante da aplicação da CQT, com $b = 36$, $f_{min} = 55$, 257 bandas de frequência, é ilustrado na Figura 9(b).

Figura 9 – Forma de onda do sinal e respectivo espectrograma da CQT.



(a) Forma de onda. **(b) Espectrograma da CQT.**

Fonte: Autoria própria.

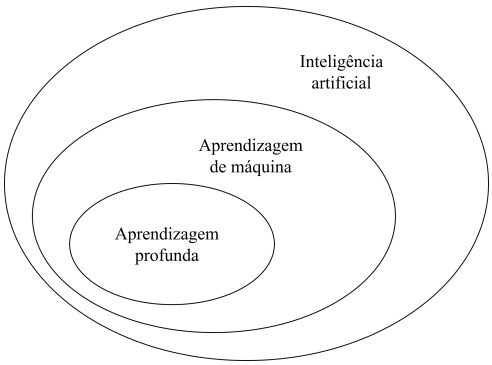
2.3 INTELIGÊNCIA ARTIFICIAL

A inteligência artificial (AI) pode ser definida como um esforço para a automatização de tarefas intelectuais que normalmente exigiriam inteligência humana (CHOLLET, 2021).

A AI engloba diversas áreas de atuação, estando presente, por exemplo, em medicina, indústria automotiva, finanças, atendimento ao cliente, *marketing*, segurança digital, redes sociais e jogos eletrônicos.

Os termos inteligência artificial, aprendizagem de máquina (*machine learning*) e aprendizagem profunda (*deep learning*), apesar de estarem diretamente relacionados, apresentam significados diferentes. Enquanto a aprendizagem de máquina é uma subárea da inteligência artificial, a aprendizagem profunda é uma abordagem específica da aprendizagem de máquina (KAMATH *et al.*, 2019). A Figura 10 ilustra essa relação.

Figura 10 – Inteligência artificial, aprendizagem de máquina e aprendizagem profunda.



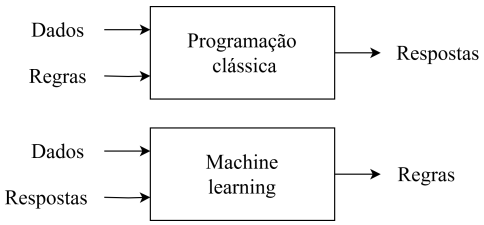
Fonte: Autoria própria.

2.3.1 Aprendizagem de máquina

De acordo com Raschka (2015), a aprendizagem de máquina é um subcampo da inteligência artificial que envolve o desenvolvimento de algoritmos que podem aprender a partir de dados e fazer previsões, sem depender de programação explícita. Ao utilizar grandes conjuntos de dados, a aprendizagem de máquina permite extrair conhecimento dos dados e aprimorar continuamente os modelos preditivos. Essa abordagem oferece uma maneira mais eficiente de analisar grandes quantidades de informações e tomar decisões orientadas por dados.

A abordagem da aprendizagem de máquina difere fundamentalmente da programação clássica, como mostrado na Figura 11. Na programação clássica, os desenvolvedores escrevem algoritmos específicos, criados com base em regras e lógicas pré-definidas, para realizar tarefas ou tomar decisões. Em *machine learning*, em vez de programar explicitamente as regras, os algoritmos são treinados em conjuntos de dados para aprender a partir dos exemplos (CHOLLET, 2021). O modelo é alimentado com dados de entrada e as saídas correspondentes, permitindo que ele identifique padrões e relacionamentos nos dados.

Figura 11 – O paradigma da aprendizagem de máquina.



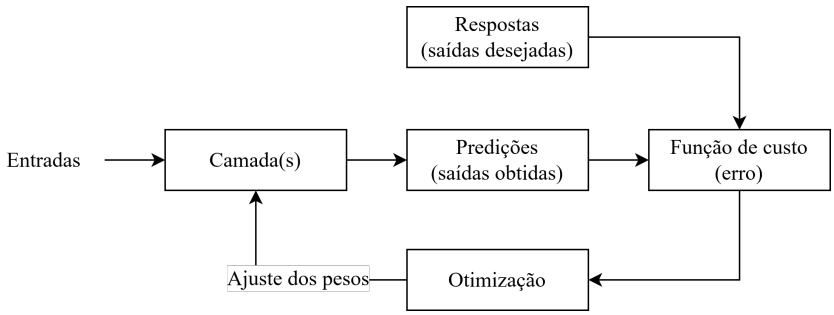
Fonte: Autoria própria.

Os sistemas de aprendizagem podem ser classificados quanto ao tipo de supervisão que eles possuem. No processo de treinamento supervisionado (Figura 12), o modelo é alimentado com exemplos rotulados, ou seja, os dados de entrada juntamente com as respostas (saídas esperadas) correspondentes (GÉRON, 2022).

Uma rede neural é composta de uma ou mais camadas de neurônios artificiais (unidades) que efetuam operações sobre os dados de entrada e extraem representações (CHOLLET, 2021). Os pesos são parâmetros ajustáveis associados às conexões entre os neurônios nas camadas do modelo. A função de custo, também chamada de função de erro, é usada para medir o erro entre as predições do modelo e as saídas desejadas.

Durante o treinamento, o modelo recebe os dados de entrada e calcula as previsões com base nos pesos iniciais, definidos aleatoriamente. A função de custo é então utilizada para avaliar o quão bem as predições correspondem às respostas esperadas. Os pesos são atualizados iterativamente com o uso algoritmos de otimização, como o gradiente descendente, para reduzir o valor da função de custo (STEVENS *et al.*, 2020).

Figura 12 – O processo de aprendizagem supervisionada.



Fonte: Autoria própria.

Inspirado pelo neurônio biológico encontrado no cérebro humano, o neurônio artificial, também conhecido como perceptron, é um componente básico de uma rede neural artificial. É uma entidade composta de uma transformação linear da entrada com os pesos seguida de uma função de ativação não-linear (STEVENS *et al.*, 2020).

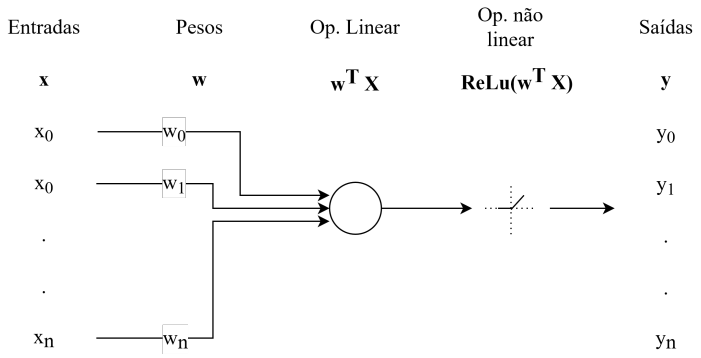
Matematicamente, a função do neurônio artificial pode ser escrita através de (10) (KAMATH *et al.*, 2019). A saída é representada por y, com x representando o vetor de entrada, w representando o vetor de pesos, b é o termo de viés (*bias*) e f é a função de ativação não-linear aplicada ao produto das entradas e pesos

$$y = f(\mathbf{x}^T\mathbf{w} + b) \tag{10}$$

Essa operação é ilustrada na Figura 13, com a função de ativação ReLU (*Rectified Linear Unit*), que aplica a função identidade quando a entrada é positiva e retorna zero quando a entrada é negativa: $f(x) = max(0,x)$.

A atualização dos pesos pode ser realizada iterativamente através do método de otimização de descida em gradiente com momento (*momentum*). O processo, mostrado no Algoritmo 1, visa diminuir a função de custo e consiste em atualizar os pesos da rede neural seguindo a direção oposta do gradiente da função de perda em relação aos pesos. Adicionalmente, o algoritmo utiliza a média móvel exponencial dos gradientes, acumulando uma fração do gradiente anterior

Figura 13 – O modelo de um neurônio artificial.



Fonte: Autoria própria.

e usando-a para influenciar a direção da atualização atual, podendo acelerar a convergência do processo.

Algoritmo 1 – Descida em gradiente com momento.

Inicializar os pesos w aleatoriamente
Definir a taxa de aprendizado α (ex: 0,001)
Definir o parâmetro β (ex: 0,9)
Inicializar o vetor de primeiro momento como zero: $m_{t=0} \leftarrow 0$
Definir o número de épocas N
Definir a função de perda L
para época de 1 até N **faça**
    para cada lote de exemplos de treinamento (x, y) **faça**
        $t \leftarrow t + 1$
        Calcular a saída prevista: $\hat{y} \leftarrow f(x \cdot w_{t-1} + b)$
        Calcular a perda entre as saídas previstas e reais: $L(\hat{y}, y)$
        Calcular o gradiente da função de perda em relação aos pesos: $\nabla L(w_{t-1})$
        Atualizar o primeiro vetor de momento: $m_t \leftarrow \beta \cdot m_{t-1} + (1 - \beta) \cdot \nabla L(w_{t-1})$
        Atualizar os pesos: $w_t \leftarrow w_{t-1} - \alpha \cdot m_t$
    finaliza para
finaliza para

O AdamW (LOSHCHILOV; HUTTER, 2017), ilustrado no Algoritmo 2, é uma técnica mais complexa de otimização baseada na descida em gradiente. O método utiliza dois momentos adaptativos para ajustar a taxa de aprendizado durante o treinamento, incluindo, além das médias móveis exponenciais, os valores quadráticos dos gradientes passados. O algoritmo também incorpora um termo de decaimento de pesos para auxiliar na regularização do modelo.

A função de custo (ou perda) de entropia cruzada (*cross entropy loss*) é frequentemente utilizada em problemas de classificação. A fórmula, que quantifica a diferença entre duas distribuições de probabilidade, pode ser descrita por

$$L(\hat{y}, y) = -\sum_i y_i \cdot \ln(\hat{y}_i) \tag{11}$$

Algoritmo 2 – AdamW.

Inicializar os pesos w aleatoriamente
Definir a taxa de aprendizado inicial α (ex: 0,001)
Definir os parâmetros: β_1 (ex: 0,9), β_2 (ex: 0,999), ϵ (ex: 1×10^{-8}), $\lambda \in \mathbb{R}$
Inicializar o vetor de primeiro momento m como zero: $m_{t=0} \leftarrow 0$
Inicializar o vetor de segundo momento v como zero: $v_{t=0} \leftarrow 0$
Inicializar o fator de programação: $\eta_{t=0} \in \mathbb{R}$
Definir o número de épocas N
Definir a função de perda L
para época de 1 até N **faça**
    para cada lote de exemplos de treinamento (x, y) **faça**
        $t \leftarrow t + 1$
        Calcular a saída prevista: $\hat{y} \leftarrow f(x \cdot w_{t-1} + b)$
        Calcular a perda entre as saídas previstas e reais: $L(\hat{y}, y)$
        Calcular o gradiente da função de perda em relação aos pesos: $\nabla L(w_{t-1})$
        Atualizar o vetor de primeiro momento: $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot \nabla L(w_{t-1})$
        Atualizar o vetor de segundo momento: $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot (\nabla L(w_{t-1}))^2$
        Corrigir o vetor de primeiro momento: $\hat{m_t} \leftarrow \frac{m_t}{1-\beta_1^t}$
        Corrigir o vetor de segundo momento: $\hat{v_t} \leftarrow \frac{v_t}{1-\beta_2^t}$
        Atualizar o fator de programação: η_t
        Atualizar os pesos: $w_t \leftarrow w_{t-1} - \eta_t \cdot (\alpha \cdot \frac{\hat{m_t}}{\sqrt{\hat{v_t}}+\epsilon} + \lambda \cdot w_{t-1})$
    finaliza para
finaliza para

onde, para uma determinada amostra do *dataset*, y_i é o valor de probabilidade real para a classe i e $\hat{y}_i$ é o valor da probabilidade prevista pelo modelo para a classe i.

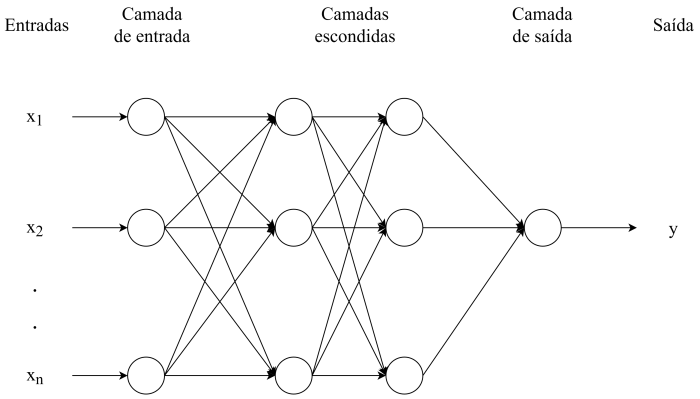
2.3.2 Aprendizagem profunda

A aprendizagem profunda, uma subárea da aprendizagem de máquina, é uma abordagem que enfatiza o aprendizado de camadas sucessivas de representações cada vez mais significativas a partir dos dados, envolvendo um grande número de camadas que são aprendidas automaticamente a partir dos dados de treinamento, permitindo que o modelo capture características complexas e tome decisões mais precisas (CHOLLET, 2021).

Redes totalmente conectadas (redes neurais densas) são compostas de neurônios interconectados. Nesse tipo de rede, as saídas de uma camada de neurônios são usadas como entradas para a camada seguinte. Uma rede totalmente conectada é composta de uma camada de entrada, uma ou mais camadas ocultas ou intermediárias e uma camada de saída (GÉRON, 2022), conforme ilustrado na Figura 14. Quando o número de camadas ocultas é amplo, a rede é considerada profunda.

Uma rede neural convolucional é um tipo de rede neural profunda projetada para o processamento de dados bidimensionais, como imagens. As CNNs têm sido responsáveis por avanços significativos em tarefas de visão computacional, como reconhecimento de imagens e

Figura 14 – Rede totalmente conectada.



Fonte: Autoria própria.

detecção/segmentação de objetos. Diferentemente das redes totalmente conectadas, as redes neurais convolucionais são compostas por camadas específicas que aplicam operações de convolução e *pooling*.

Enquanto as camadas totalmente conectadas aprendem padrões globais em seu espaço de entrada, as camadas convolucionais detectam padrões locais com hierarquia espacial e invariância à translação (CHOLLET, 2021). A hierarquia espacial implica que as primeiras camadas de convolução detectem padrões locais menores e as posteriores detectem padrões maiores formados pelas características das camadas anteriores. A invariância à translação significa que um padrão detectado em uma determinada posição na imagem pode ser reconhecido em qualquer outro local.

A convolução discreta 2D é definida para uma imagem bidimensional como o produto escalar de uma matriz de pesos, também denominada filtro ou *kernel*, com cada vizinhança na imagem de entrada (STEVENS *et al.*, 2020). Em outras palavras, a matriz de pesos é deslocada sobre a imagem de entrada e o valor resultante na posição correspondente da imagem de saída é calculado como a soma dos produtos.

A fórmula da convolução discreta 2D pode ser definida através de (KAMATH *et al.*, 2019)

$$y[i,j] = \sum_{n}\sum_{m} h[m,n] \cdot x[i-m, i-n] \tag{12}$$

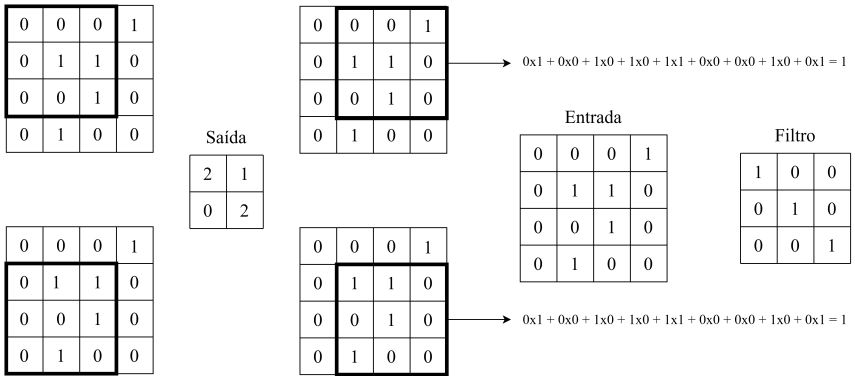
onde x é a imagem de entrada, h é o filtro de tamanho $n \times m$ e y é imagem de saída.

Dois parâmetros importantes utilizados nas camadas convolucionais e de *pooling* são o

padding e o *stride*. O *padding* refere-se à adição de bordas ao redor da imagem de entrada antes de realizar alguma operação para controlar o tamanho da saída da camada. O *stride* é o passo do deslocamento do filtro sobre a imagem. Um *stride* maior resulta em uma subamostragem mais agressiva.

A Figura 15 apresenta um exemplo para uma operação de convolução entre um filtro de tamanho 3x3 com deslocamento unitário e uma imagem de entrada de tamanho 4x4, resultando em uma imagem de saída de tamanho 2x2.

Figura 15 – A operação de convolução.

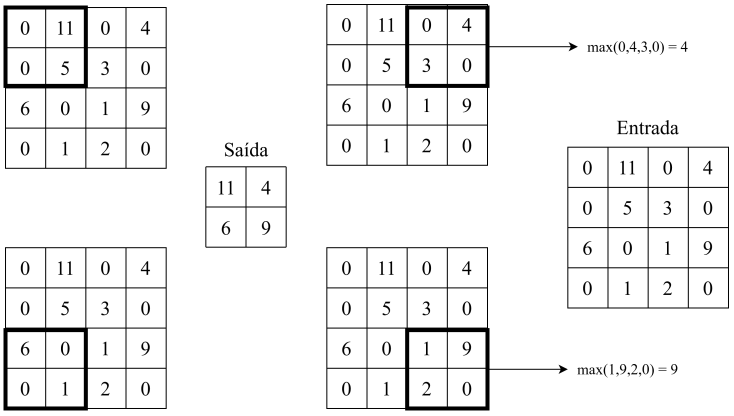


Fonte: Autoria própria.

As camadas de *pooling* têm o objetivo de reduzir o tamanho da imagem de entrada através de um processo de subamostragem, consequentemente diminuindo o número de parâmetros do modelo. Um neurônio de *pooling* não possui pesos. O tipo mais comum de camada de *pooling* é a que utiliza a função *max pooling*, onde apenas o valor máximo de cada campo receptivo é propagado para a próxima camada (GÉRON, 2022).

A Figura 16 apresenta um exemplo para uma operação de *max pooling* entre um filtro de tamanho 2x2 com deslocamento de duas unidades e uma imagem de entrada de tamanho 4x4, resultando em uma imagem de saída de tamanho 2x2.

Figura 16 – A operação de *max pooling*.



Fonte: Autoria própria.

3 TRABALHOS RELACIONADOS

Diversos métodos foram propostos para detecção de edições em registros de áudios. A maioria dos estudos encontrados requer, como etapa de pré-processamento, a extração manual de vetores de características, o que pode ser demorado, suscetível a erros e limitado pelo conhecimento prévio.

Recentemente, alguns trabalhos que utilizam arquiteturas de aprendizagem profunda para autenticação de áudio foram publicados. A utilização de *deep learning* (DL) possibilita a extração automática das *features* e a captura de características complexas e abstratas em diferentes níveis. As redes neurais profundas podem aprender com uma ampla gama de dados e se adaptar a diferentes tipos de edições, tornando os métodos baseados em DL mais flexíveis.

A seguir são apresentados alguns dos métodos encontrados, separados conforme o procedimento de extração de *features* para classificação.

3.1 MÉTODOS COM EXTRAÇÃO MANUAL DE *FEATURES*

Shi e Ma (2011) desenvolveram um método para detectar interpolações utilizando a decomposição em valores singulares (*Singular Value Decomposition* - SVD) para analisar o áudio e identificar alterações estatísticas nas dependências lineares entre os pontos do sinal. A SVD é uma técnica matemática utilizada para decompor uma matriz em componentes principais. Um vetor de características é gerado a partir dessa análise, contando o número médio de valores singulares iguais a zero e descrevendo, portanto, as alterações estatísticas resultantes do processo de interpolação.

O estudo de Pan *et al.* (2012) propõe um método para detectar interpolações em sinais de áudio digitais utilizando diferenças anormais nos níveis de ruído local como indicadores de adulteração. A estimativa dos níveis de ruído é baseada na observação de que os sinais de áudio têm uma curtose constante em determinadas frequências. A curtose é uma medida estatística que descreve a forma da distribuição de um conjunto de dados, medindo o quão "achatada"ou "pontiaguda"a distribuição é em relação a uma distribuição normal. Os autores calculam a variância do ruído minimizando uma função objetivo e identificam trechos suspeitos de áudio com base na inconsistência nos níveis de ruído.

Zhao *et al.* (2014) propõem um método para autenticar áudios e identificar segmentos

adulterados. Eles utilizam as magnitudes da resposta ao impulso do canal acústico (*Room Impulse Response* - RIR) e do ruído local como assinatura ambiental. O método extrai essas magnitudes, calcula a correlação entre o áudio questionado e o de referência e utiliza um limiar ótimo para identificar segmentos adulterados. Também é aplicada uma etapa de refinamento usando informações dos quadros adjacentes.

Hua *et al.* (2016) exploraram a detecção de adulteração de áudio baseada na frequência da rede elétrica (*Electrical Network Frequency* - ENF), que é uma assinatura natural presente em muitas gravações de áudio e é usualmente utilizada como carimbo de tempo. Os autores introduzem o conceito de "*absolute-error-map* - AEM", que representa um mapa de erros absolutos entre os sinais ENF obtidos da gravação de áudio de teste e do banco de dados. Com base na análise do AEM, os autores desenvolvem dois algoritmos para verificar o carimbo de tempo e detectar adulterações, como inserções, supressões e interpolações. O primeiro algoritmo é baseado em busca exaustiva e medição, enquanto o segundo utiliza a técnica de erosão de imagem. O sistema verifica se há adulteração nos dados de teste e, caso não detecte, fornece informações de carimbo de tempo. No caso de detecção, identifica o tipo de adulteração e a região afetada.

No trabalho de Capoferri *et al.* (2020), a detecção de interpolações é realizada através da identificação de inconsistências no tempo de reverberação ao longo do áudio. A reverberação é um fenômeno acústico que ocorre quando o som é refletido nas superfícies de um ambiente. O tempo de reverberação é o tempo que leva para o som decair em certa intensidade após a fonte sonora ter sido desligada. Os autores aproveitam a ideia de que gravações distintas podem ser feitas em ambientes diferentes, que geralmente são caracterizados por traços de reverberação distintos. Se o tempo de reverberação mudar abruptamente de um instante para outro, o áudio é marcado como manipulado e o ponto de edição é estimado.

3.2 MÉTODOS COM APRENDIZAGEM PROFUNDA

No estudo de Jadhav *et al.* (2019), o espectrograma da STFT é fornecido diretamente como dado de entrada para a rede neural convolucional para detecção de interpolações em áudios. A rede proposta é composta de duas camadas convolucionais seguidas de camadas de *batch normalization* e função de ativação ReLU, uma camada de *max pooling*, uma camada totalmente conectada e uma camada de classificação com função softmax.

Ustubioglu *et al.* (2022) utilizaram o mel-espectrograma da STFT como o dado de

entrada da CNN para a detecção de réplicas (*copy-move*) em áudios. A rede proposta é composta de quatro camadas convolucionais com função de ativação ReLU, três camada de *dropout*, uma camada de *max pooling*, uma camada totalmente conectada e uma camada de classificação com função softmax.

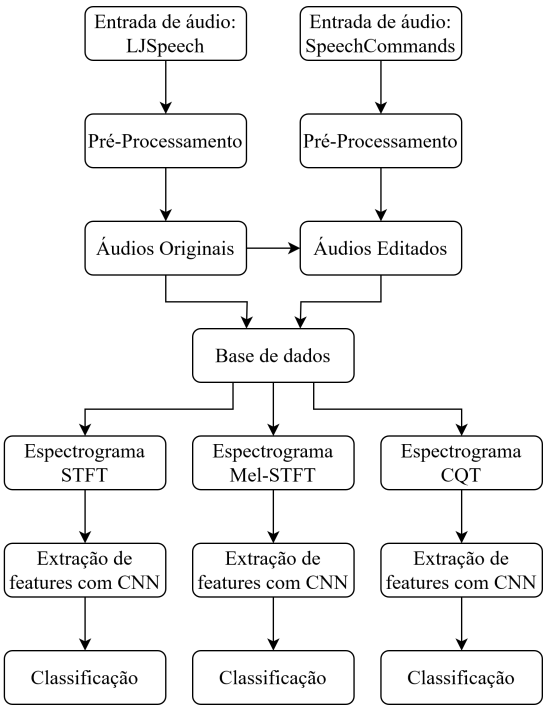
A arquitetura *Transformer* foi utilizada no trabalho de Moussa *et al.* (2022) para a detecção de interpolações em áudios. Essa arquitetura é um modelo de aprendizado de máquina usado principalmente para tarefas de processamento de linguagem natural (NLP) baseada em mecanismos de atenção. A rede proposta no trabalho opera em três diferentes representações de entrada: mel-espectrograma, gráfico dos coeficientes mel-cepstrais (*Mel Frequency Cepstral Coefficients* - MFCCs) e gráfico do centroide espectral.

Zeng e Wu (2022) utilizaram o modelo de rede neural residual ResNet-18 para detecção de interpolações em áudios. A rede possui 18 camadas, incluindo 17 camadas convolucionais e uma camada totalmente conectada. A arquitetura também inclui blocos residuais, que são caminhos adicionais que permitem que as informações do início da rede sejam transmitidas diretamente para camadas posteriores. Os autores utilizaram como o dado de entrada o gráfico dos coeficientes mel-cepstrais, o gráfico dos coeficientes lineares cepstrais (*Linear Frequency Cepstral Coefficients* - LFCC) e o gráfico dos coeficientes do banco de filtros mel.

4 MATERIAL E MÉTODOS

Inicialmente, os áudios foram pré-processados e a base de dados contendo os áudios originais e editados foi gerada. Considerando que uma rede neural convolucional é aplicada no processamento e análise de dados bidimensionais, como imagens, o espectrograma dos áudios foi calculado e fornecido como o dado de entrada da rede, através de três diferentes representações no domínio tempo-frequência: a transformada de fourier de tempo curto (STFT) na escala linear, STFT na escala mel e a transformada Q constante (CQT). Um esquema ilustrativo dos procedimentos é mostrado na Figura 17.

Figura 17 – Metodologia do trabalho.



Fonte: Autoria própria.

Os algoritmos foram desenvolvidos em linguagem de programação Python com a utilização de recursos computacionais disponíveis na plataforma interativa baseada em nuvem Google Colab.

4.1 PRÉ-PROCESSAMENTO E GERAÇÃO DOS ÁUDIOS EDITADOS

No presente estudo, a base de dados consistiu em 4000 áudios originais do *dataset* LJSpeech (ITO; JOHNSON, 2017) e 4000 áudios editados com a inserção de trechos de áudios do *dataset* SpeechCommands (WARDEN, 2018).

O LJSpeech (LJ) é um conjunto de 13100 áudios com duração entre 1 e 11 segundos e taxa de amostragem de 22050 Hz, onde um único orador realiza a leitura de trechos de livros de não-ficção, com um número médio de 17 palavras por áudio. O SpeechCommands (SC) contém 105830 áudios de duração aproximada de 1 segundo e taxa de amostragem de 16000 Hz, consistindo em 35 comandos de voz emitidos por diferentes falantes. Os dois conjuntos de dados contém informações adicionais, como a transcrição dos áudios e o valor da taxa de amostragem.

Os áudios originais foram extraídos do LJSpeech. Para a realização dos experimentos, foram selecionados os primeiros segundos de cada áudio, com variação da janela de seleção entre 4 e 8 segundos, desconsiderando-se os registros que apresentam duração inferior a 8 segundos. Os 4000 sinais originais selecionados foram reamostrados para 16 kHz e normalizados.

Além da comparação entre as diferentes representações tempo-frequência, a duração dos áudios foi variada para visualizar o efeito da alteração da proporção do trecho editado em relação à extensão total do sinal. A escolha da duração dos áudios foi realizada a fim de garantir que o mesmo conjunto de arquivos fosse processado em cada execução do algoritmo. Dos 13100 áudios do *dataset*, apenas 4067 possuem duração mínima de 8 segundos.

Para a geração dos áudios editados através de interpolação, um subconjunto de 4000 áudios do *dataset* SpeechCommands foi extraído, sendo removidos os trechos de silêncio de cada sinal, com posterior normalização dos valores. Os registros extraídos e processados do SpeechCommands, apresentando duração média aproximada de 0,40s, foram copiados e inseridos em uma determinada posição nos áudios originais.

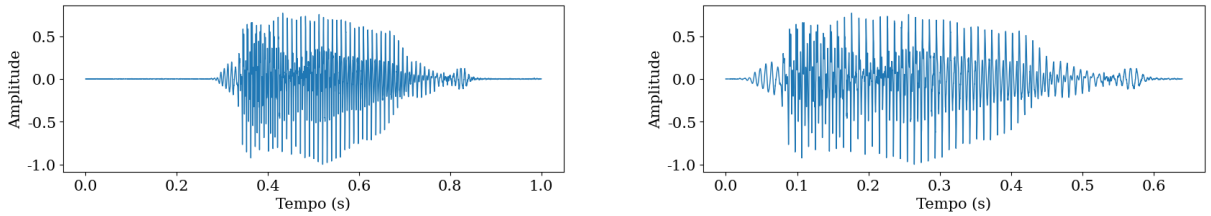
Para a padronização dos dados, a normalização dos áudios dos dois *datasets* foi realizada através da técnica *z-score*, que transforma os dados em uma distribuição com média μ_x zero e desvio padrão σ_x unitário

$$x_{\text{norm}}[n] = \frac{x[n] - \mu_x}{\sigma_x} \tag{13}$$

A determinação da posição da inserção dos trechos dos áudios do *dataset* SC nos áudios originais foi baseada no processo ilustrado na Figura 19. Os áudios do *dataset* LJ foram segmentados com um algoritmo de detecção de atividade sonora e a duração de cada trecho

obtido foi comparada com a duração de um áudio do *dataset* SC. Após a identificação do segmento com o tamanho mais próximo, o áudio do *dataset* SC é inserido na posição de início do segmento identificado. Também foi implementada uma lógica para garantir que o procedimento de edição não resultasse em um aumento da duração do áudio.

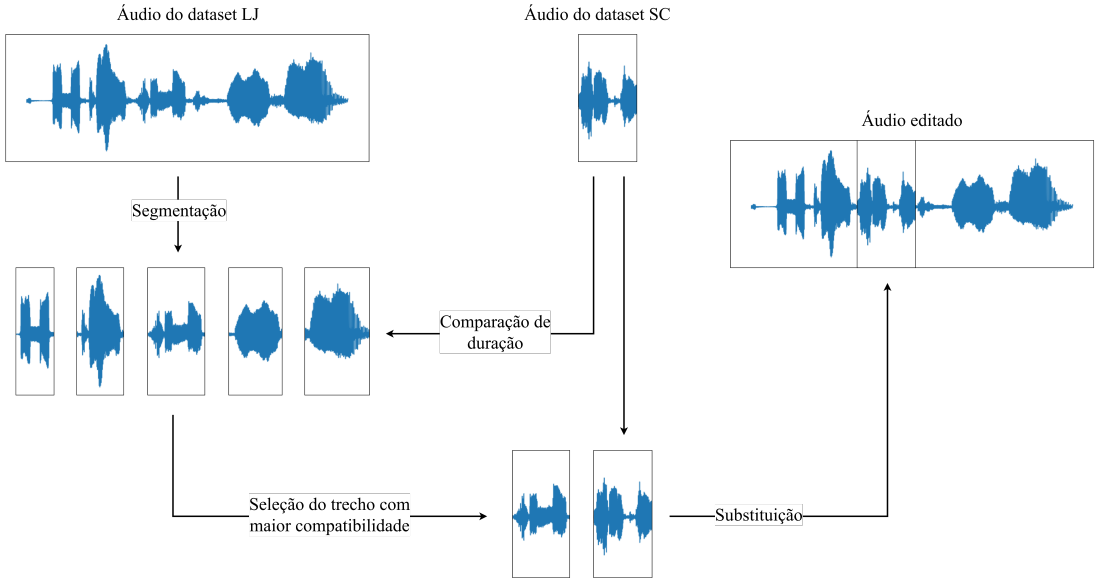
Figura 18 – Procedimento de remoção de silêncio de um sinal do *dataset* SC.



(a) Forma de onda original. **(b) Forma de onda processada.**

Fonte: Autoria própria.

Figura 19 – Ilustração do procedimento de geração dos áudios editados.



Fonte: Autoria própria.

A segmentação das locuções dos áudios do *dataset* LJ e a remoção dos trechos de silêncio dos áudios do *dataset* SC foram realizadas com base em um algoritmo de detecção de atividade sonora. A função utiliza um limiar de energia para identificar as partes do sinal de áudio que contêm trechos de silêncio. O valor quadrático médio do sinal (RMS), expresso em (14), foi calculado para uma janela deslocada no tempo e comparado com o ponto máximo do sinal

$$RMS = \frac{1}{N}\sum_{n=0}^{N-1} x^2(n) \tag{14}$$

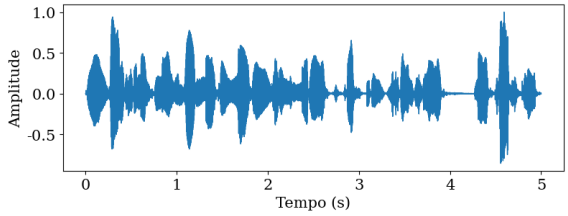
Para a segmentação dos áudios do *dataset* LJ e para a remoção dos trechos de silêncio no início e fim dos áudios do *dataset* SC, o tamanho da janela foi configurado em $N = 2048$, com deslocamento da janela de 512 amostras. O limiar abaixo da referência para considerar o trecho como silêncio foi configurado em 20 dB.

O *dataset* LJ contém áudios com boa razão sinal-ruído (SNR - *Signal-to-Noise Ratio*). O *dataset* SC apresenta alta variabilidade referente à qualidade dos áudios, contendo algumas amostras com elevado ruído de fundo. O algoritmo de atividade sonora descrito anteriormente, apesar de ter funcionado bem para a segmentação de voz dos áudios do LJSpeech, não foi efetivo para isolar trechos de voz em áudios com baixa SNR do SpeechCommands. A solução encontrada para o problema descrito foi selecionar somente os áudios do SpeechCommands que, após serem processado pelo algoritmo na tentativa de remoção de trechos não vozeados, não excedessem a duração de 0,6s.

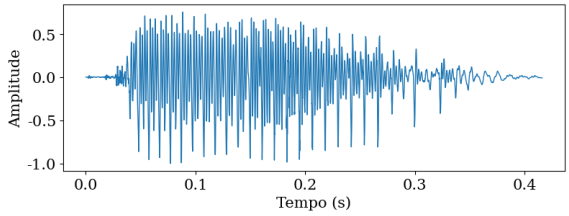
Os registros do LJSpeech foram gerados através de segmentação automática da gravação, não apresentando, portanto, diferenças significativas de conteúdo espectral. Foram realizados testes com o modelo de rede neural desenvolvido no presente trabalho para detectar edições efetuadas somente entre áudios do referido *dataset* (réplica ou *copy-move*), não se obtendo êxito. A rede não foi capaz de identificar os áudios editados quando não existem alterações no padrão do ruído de fundo.

Na Figura 20 são apresentadas formas de onda de amostras dos *datasets* utilizados após as operações de pré-processamento e de um exemplo do sinal interpolado resultante.

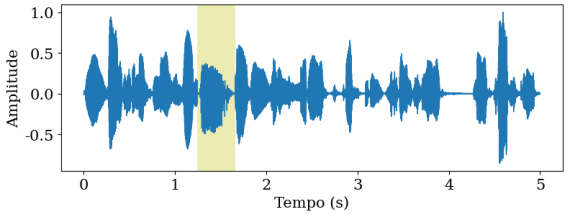
Figura 20 – Exemplo de geração de um sinal editado a partir de amostras dos *datasets* utilizados.



(a) Forma de onda de um sinal do *dataset* LJ.



(b) Forma de onda de um sinal do *dataset* SC.

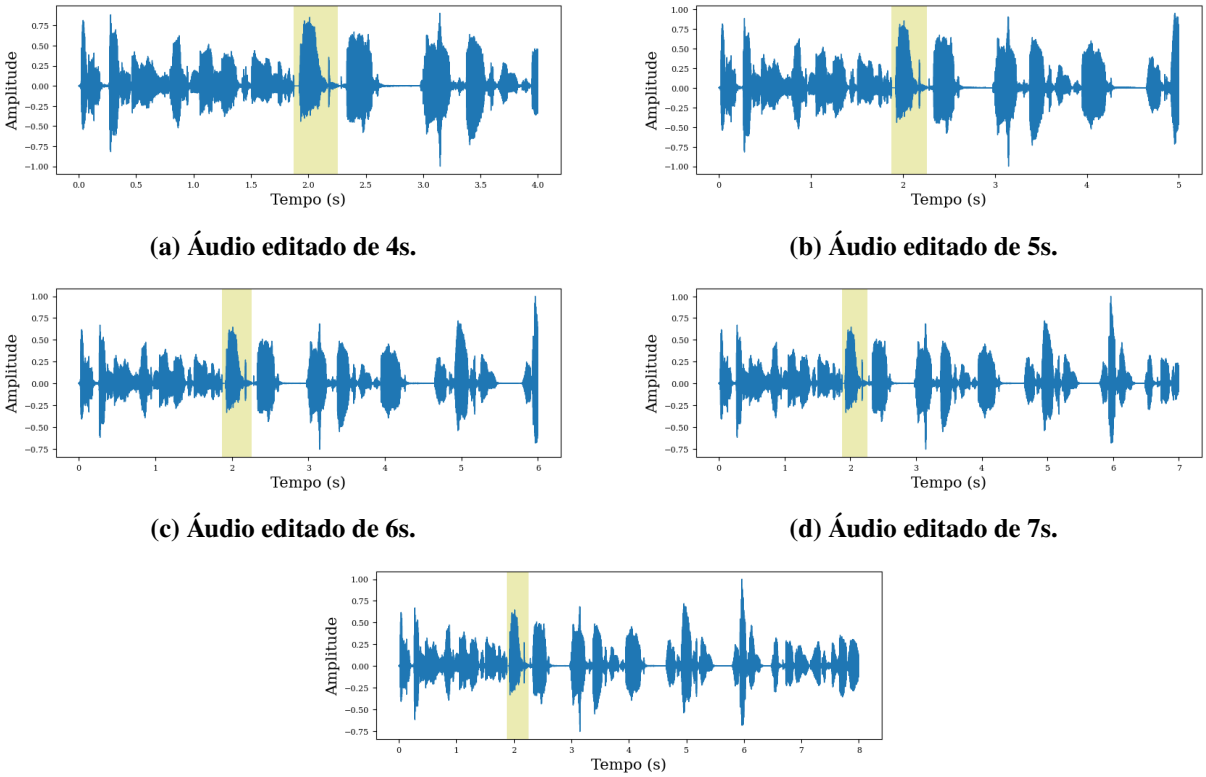


(c) Forma de onda do áudio editado.

Fonte: Autoria própria.

De modo análogo ao apresentado na Fig. 21, a Fig. 23 ilustra, nos espectrogramas da STFT linear, a variação da proporção do trecho editado em relação à extensão total do áudio.

Figura 21 – Visualização da proporção do trecho editado nas formas de onda.



(a) Áudio editado de 4s.

(b) Áudio editado de 5s.

(c) Áudio editado de 6s.

(d) Áudio editado de 7s.

(e) Áudio editado de 8s.

Fonte: Autoria própria.

4.2 REPRESENTAÇÃO TEMPO-FREQUÊNCIA

Os espectrogramas da STFT dos áudios foram gerados com a utilização das bibliotecas Torchaudio e Numpy. Foi selecionado o janelamento de Hann de tamanho $N = 512$ e deslocamento da janela com passo de $N/2$. As frequências foram dispostas de forma linear e na escala mel (mel-espectrograma) com 257 bandas.

Os espectrogramas da CQT dos áudios foram gerados com a utilização das bibliotecas nnAudio e Numpy, com 257 bandas e $b = 36$ filtros dentro de cada oitava. O deslocamento da janela de comprimento variável foi realizado com passo de $N/2$. A frequência mínima foi configurada em $f_{min} = 55$ Hz.

A Tabela 1 apresenta a dimensão dos espectrogramas gerados em função da variação da duração dos áudios utilizados.

Algoritmo 3 – Geração dos áudios editados.

Entrada: *audios_LJ*, *audios_SC*
audios_editados ← lista vazia
para i **de** 1 **até tamanho de** *audios_LJ* **faça**
    audio_LJ ← *audios_LJ*[i]
    audio_SC ← *audios_SC*[i]
    segmentos_audio_LJ ← segmentação(*audio_LJ*)
    menor_diferença ← infinito
    i_menor_diferença ← -1
    para $j, (inicio, fim)$ **em** *segmentos_audio_LJ* **faça**
        tamanho_segmento ← *fim* - *inicio*
        dif_tamanho ← valor absoluto(tamanho de *tamanho_segmento* - tamanho de *audio_SC*)
        se *dif_tamanho* **<** *menor_diferença* **e** (tamanho de *audio_SC* + *inicio*) **<** tamanho de *audio_LJ* **então**
            menor_diferença ← *dif_tamanho*
            i_menor_diferença ← *j*
        finaliza se
    finaliza para
    inicio, fim ← *segmentos_audio_LJ[i_menor_diferença]*
    audio_editado ← *audio_LJ*
    audio_editado[inicio:inicio + tamanho_de_audio_SC] ← *audio_SC*
    adicionar *audio_editado* à lista de *audios_editados*
finaliza para
Saída: *audios_editados*

Dur. Áudio	Comprimento	Altura
4s	251	257
5s	313	257
6s	376	257
7s	438	257
8s	501	257

Tabela 1 – Dimensões dos espectrogramas gerados.

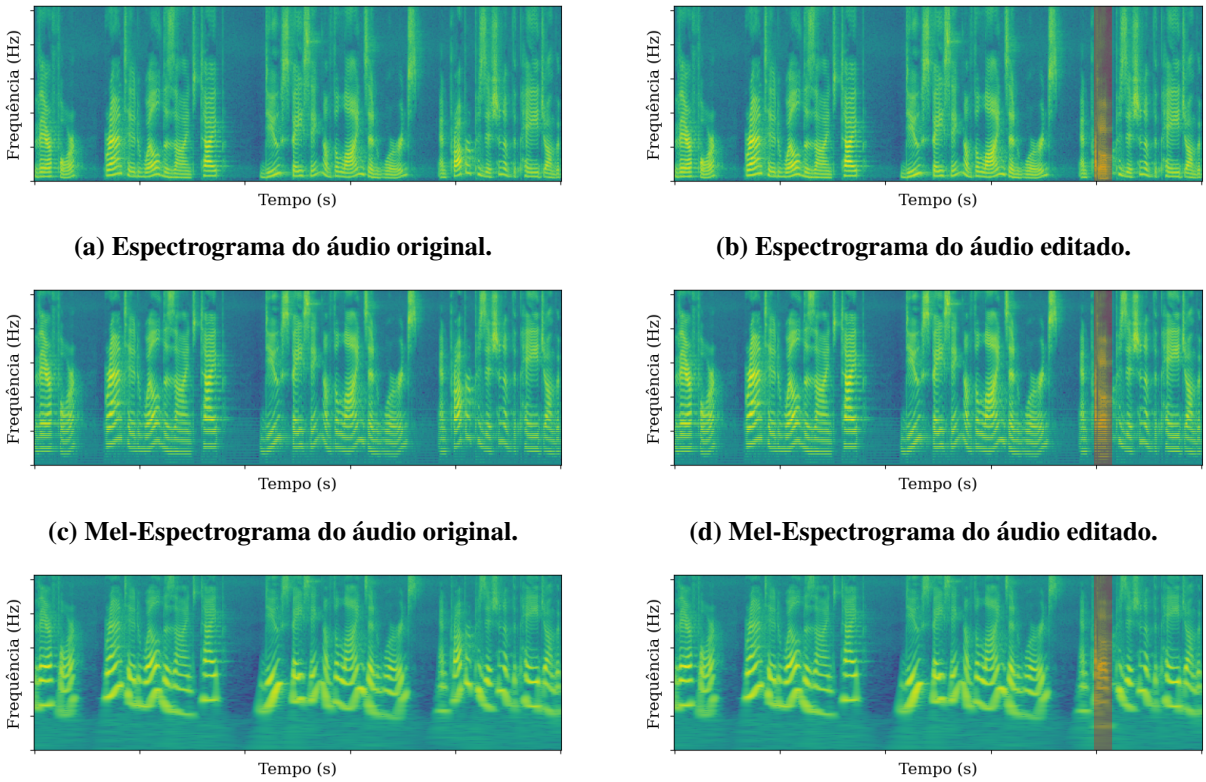
A Figura 22 apresenta os diferentes espectrogramas dos áudios originais e editados, com a indicação da localização do segmento inserido nos áudios interpolados.

A Figura 23 ilustra a variação da proporção do trecho editado em relação à extensão total do áudio. Considerando que os áudios inseridos possuem duração média de 0,40s, a proporção do segmento adulterado varia de aproximadamente 10% para os áudios de 4s até 5% para os áudios de 8s.

4.3 ARQUITETURA DA REDE NEURAL E TREINAMENTO

A arquitetura da rede neural proposta é composta pelas camadas e funções descritas a seguir. A implementação do modelo foi realizada com o uso da biblioteca PyTorch. A definição da arquitetura da rede foi baseada em projetos amplamente reconhecidos e em modelos desenvolvidos em trabalhos correlatos. No total, a rede apresenta 19949570 parâmetros.

Figura 22 – Amostras dos espectrogramas gerados com indicação do trecho de edição.



(a) Espectrograma do áudio original.

(b) Espectrograma do áudio editado.

(c) Mel-Espectrograma do áudio original.

(d) Mel-Espectrograma do áudio editado.

(e) Espectrograma da CQT do áudio original.

(f) Espectrograma da CQT do áudio editado.

Fonte: Autoria própria.

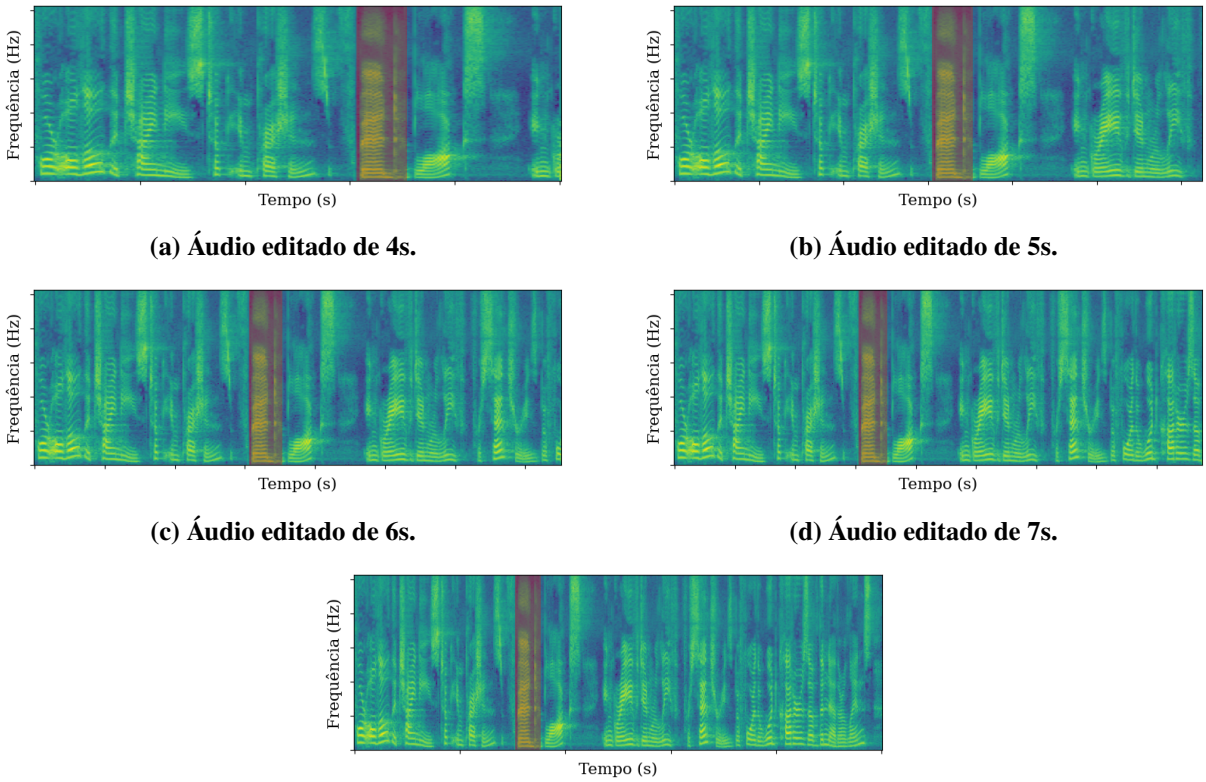
- 03 (três) camadas convolucionais com função de ativação de unidade linear retificada (ReLU). A primeira camada possui 16 filtros de tamanho 5x5 com deslocamento (*stride*) de duas unidades. A segunda e a terceira camada possuem, respectivamente, 32 e 64 filtros de tamanho 3x3 com deslocamento unitário. Todas as camadas possuem *padding* de uma unidade.

- 03 (três) camadas de agrupamento máximo (*max pooling*), após as camadas convolucionais, que incluem filtros de tamanho 2x2 com deslocamento de duas unidades.

- 04 (quatro) camadas de *dropout*, após as camadas de agrupamento e após a saída da primeira camada totalmente conectada, com parâmetro p = 0,2 (20%). O parâmetro p representa a probabilidade de desativar (zerar) aleatoriamente as saídas de um tensor durante o treinamento.

- 02 (duas) camadas totalmente conectadas (*fully connected*). A primeira camada possui 1024 unidades de saída com a função de ativação ReLU. A última representa a camada de classificação, utilizando a função de ativação exponencial normalizada (*softmax*).

Figura 23 – Visualização da proporção do trecho editado nos espectrogramas.



(a) Áudio editado de 4s.

(b) Áudio editado de 5s.

(c) Áudio editado de 6s.

(d) Áudio editado de 7s.

(e) Áudio editado de 8s.

Fonte: Autoria própria.

A utilização das camadas convolucionais permite que a rede neural capture informações locais das imagens de entrada, levando em consideração a hierarquia espacial e garantindo invariância à translação. A função de ativação ReLU introduz não-linearidade na rede, possibilitando a aprendizagem de representações mais complexas dos dados. As camadas de agrupamento máximo reduzem a dimensionalidade dos dados, mantendo as características mais importantes. O *dropout* é aplicado para evitar o *overfitting*, desativando aleatoriamente alguns neurônios durante o treinamento para regularização. Por fim, as camadas totalmente conectadas combinam as informações extraídas anteriormente para realizar a classificação final. A função softmax é aplicada na camada de saída para normalizar as probabilidades de cada classe, fornecendo uma distribuição de probabilidade sobre as classes possíveis.

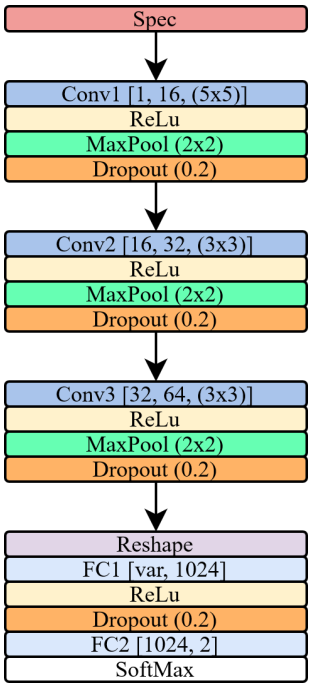
O treinamento da rede foi realizado em 50 épocas sobre o conjunto de 80% dos espectrogramas gerados, divididos em lotes de 256 (*batch size*).

Foi utilizado o algoritmo adaptativo de otimização AdamW (LOSHCHILOV; HUTTER, 2017), com taxa de aprendizagem inicial $\alpha = 0{,}001$, com a definição dos parâmetros: $\beta_1 = 0{,}9$, $\beta_2 = 0{,}999$, $\epsilon = \times 10^{-8}$, $\lambda = 0{,}01$. A função de custo de entropia cruzada (*cross-entropy loss*)

foi definida para a minimização.

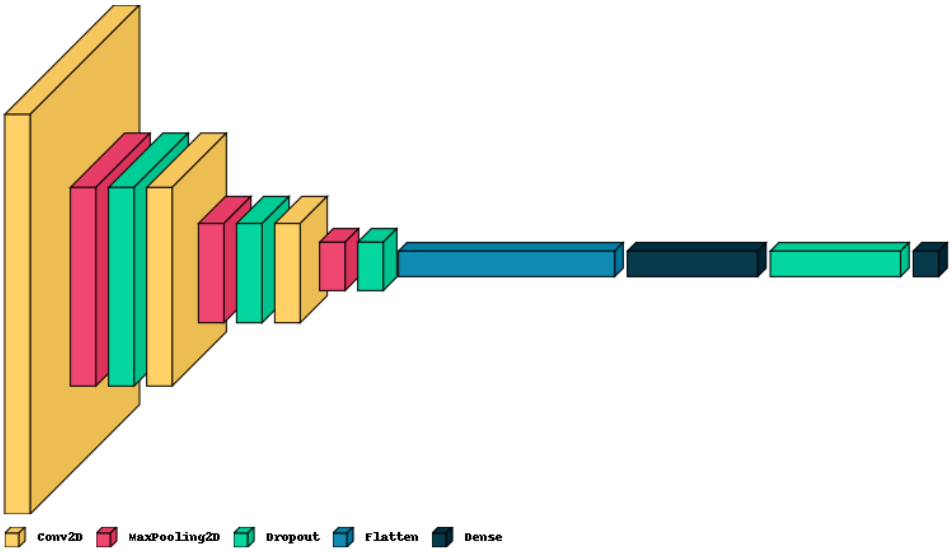
A Figura 24 apresenta um diagrama da arquitetura da rede neural proposta, com as indicações das camadas e funções que a compõem, tamanhos e dimensões dos filtros de convolução e *pooling*. A Figura 25 ilustra a visualização tridimensional das camadas.

Figura 24 – Arquitetura da rede neural proposta.



Fonte: Autoria própria.

Figura 25 – Visualização da rede neural proposta.



Fonte: Autoria própria.

4.4 MÉTRICAS DE AVALIAÇÃO

Considerando a metodologia dos trabalhos correlatos e o fato de que o estudo aborda uma tarefa de classificação binária com um *dataset* balanceado, a avaliação do desempenho dos modelos foi realizada através da métrica de acurácia, dividindo-se os acertos pelo número total de exemplos classificados

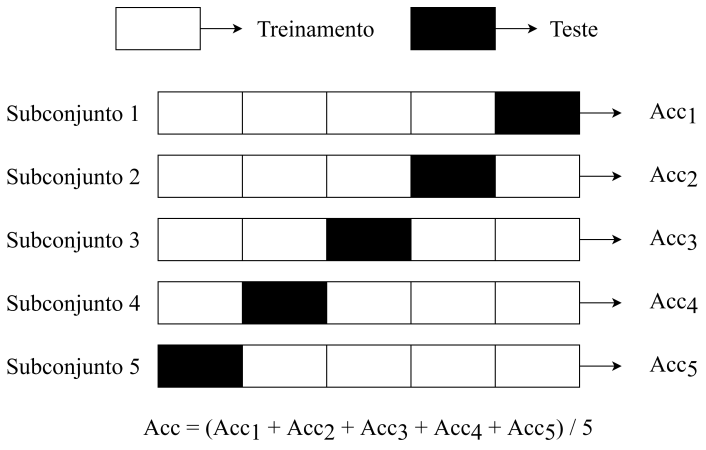
$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

onde TP (*True Positive*) é o número de exemplos positivos (editados - ocorrência do evento de edição) corretamente classificados, TN (*True Negative*) é o número de exemplos negativos (originais - ausência do evento de edição) corretamente classificados, FP (*False Positivo*) e FN (*False Negative*) correspondem aos exemplos positivos e negativos classificados incorretamente.

Com o intuito de se obter uma medida mais representativa para avaliar a capacidade de generalização do modelo a partir das diferentes entradas de dados, foi utilizada a técnica de validação cruzada *K-fold*. O procedimento consiste em dividir o conjunto total de dados em K subconjuntos (*folds*) de mesmo tamanho. O modelo é treinado K vezes e, em cada iteração, um subconjunto é utilizado para teste e $K - 1$ são utilizados para treinamento.

No presente trabalho, foram efetuadas cinco subdivisões ($K = 5$), ou seja, a cada iteração, foram utilizados 80% dos exemplos para treinamento e 20 % para teste. Ao final de cada uma das cinco execuções, foi registrado o valor da acurácia para o conjunto de teste. A acurácia média foi então utilizada como medida geral de desempenho do modelo, acompanhada do desvio padrão. O procedimento descrito está ilustrado na Figura 26.

Figura 26 – Validação cruzada (K-Fold) com K=5.

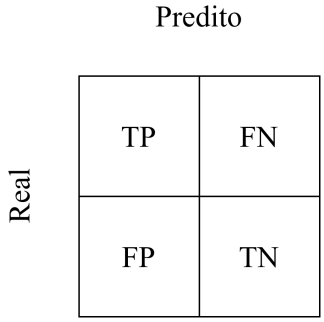


Fonte: Autoria própria.

O monitoramento do progresso de treinamento do modelo para o ajuste dos hiperparâmetros foi realizado através da visualização das curvas de acurácia e perda média após as cinco execuções de treinamento e teste.

A matriz de confusão, ilustrada na Figura 27, consiste em uma tabela que apresenta a distribuição dos exemplos classificadas corretamente e erroneamente para cada classe. Durante os procedimentos realizados, o número de exemplos classificados nos conjuntos de teste foi somado após as cinco iterações e a tabela foi disponibilizada com os valores normalizados.

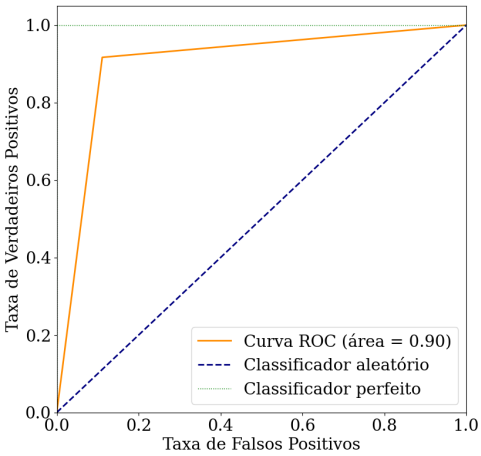
Figura 27 – Matriz de confusão.



Fonte: Autoria própria.

Para a visualização dos resultados também foi apresentada a curva ROC (*Receiver Operating Characteristic Curve*), obtida pela representação da relação entre a taxa de verdadeiros positivos $TP_R = \frac{TP}{TP+FN}$ e a taxa de falsos positivos $FP_R = \frac{FP}{FP+TN}$. A curva, mostrada na Figura 28, é uma ferramenta útil para a visualização do desempenho do classificador binário. Quanto maior a área sob a curva ROC (AUC - *Area Under the Curve*), melhor o desempenho em classificar corretamente as amostras. Os valores utilizados foram os mesmos obtidos para a geração da matriz de confusão.

Figura 28 – Curva ROC.



Fonte: Autoria própria.

5 RESULTADOS

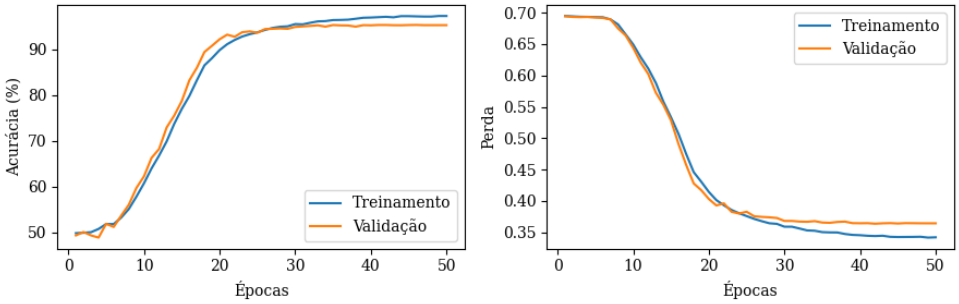
5.1 RESULTADOS OBTIDOS

A Figura 29 apresenta as curvas de acurácia média e perda média ao longo das cinco iterações para os conjuntos de treinamento e teste dos áudios de 4 segundos de duração.

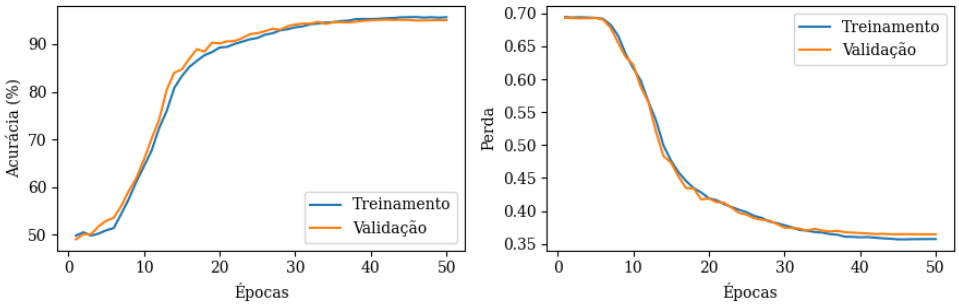
Na Figura 30 são ilustradas as matrizes de confusão normalizadas após as cinco iterações para os conjuntos de teste dos áudios de 4 segundos de duração.

A curva ROC é visualizada na Figura 31 após as cinco iterações para os conjuntos de teste dos áudios de 4 segundos de duração.

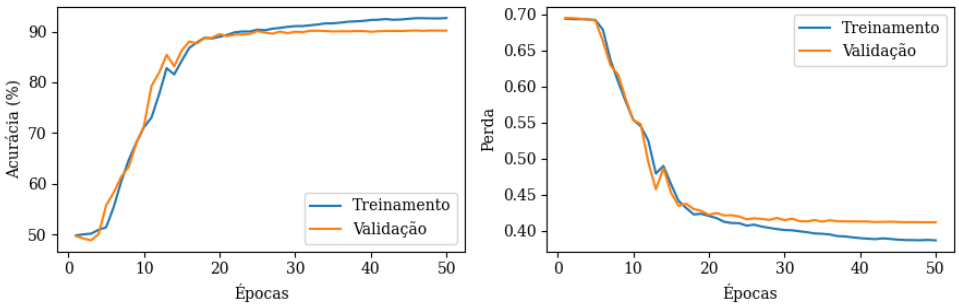
Figura 29 – Curvas de treinamento e teste referentes aos áudios de 4s.



(a) STFT.



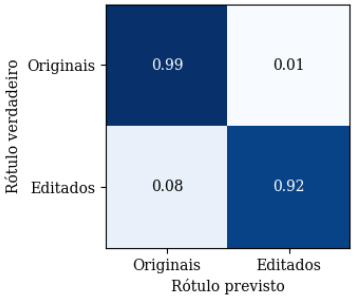
(b) Mel-STFT.



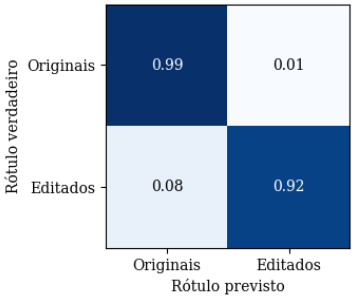
(c) CQT.

Fonte: Autoria própria.

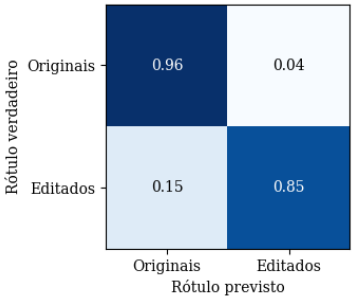
Figura 30 – Matrizes de confusão referentes aos áudios de 4s.



(a) STFT.



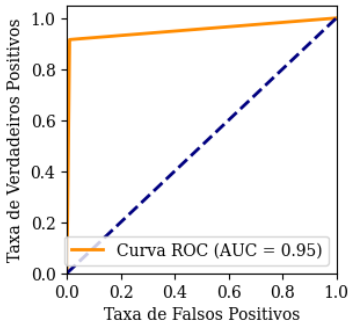
(b) Mel-STFT.



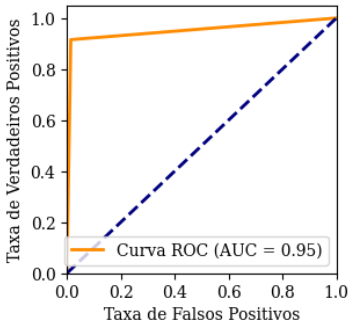
(c) CQT.

Fonte: Autoria própria.

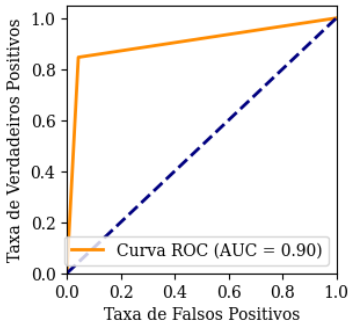
Figura 31 – Curvas ROC referentes aos áudios de 4s.



(a) STFT.



(b) Mel-STFT.



(c) CQT.

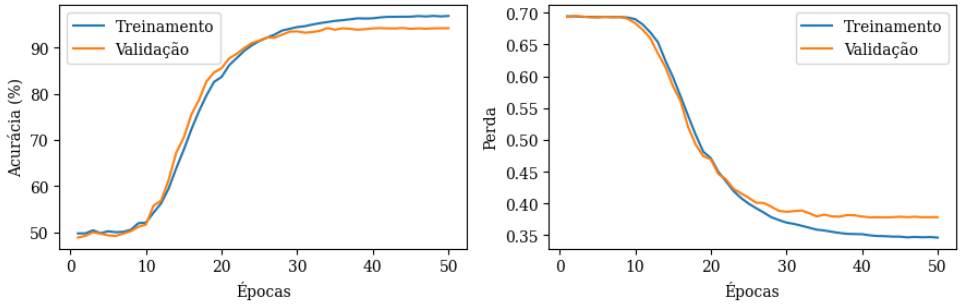
Fonte: Autoria própria.

A Figura 32 apresenta as curvas de acurácia média e perda média ao longo das cinco iterações para os conjuntos de treinamento e teste dos áudios de 5 segundos de duração.

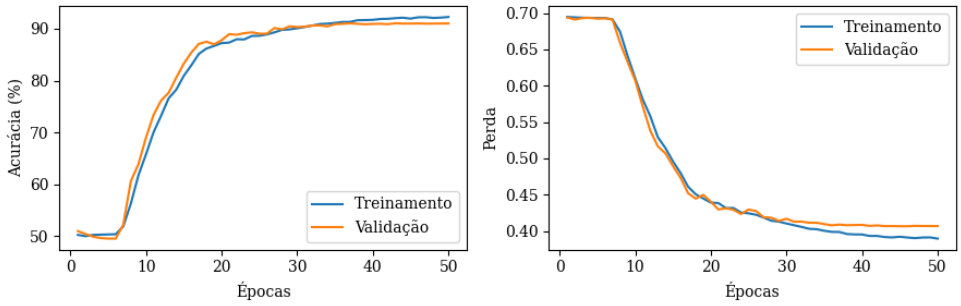
Na Figura 33 são ilustradas as matrizes de confusão normalizadas após as cinco iterações para os conjuntos de teste dos áudios de 5 segundos de duração.

A curva ROC é visualizada na Figura 34 após as cinco iterações para os conjuntos de teste dos áudios de 5 segundos de duração.

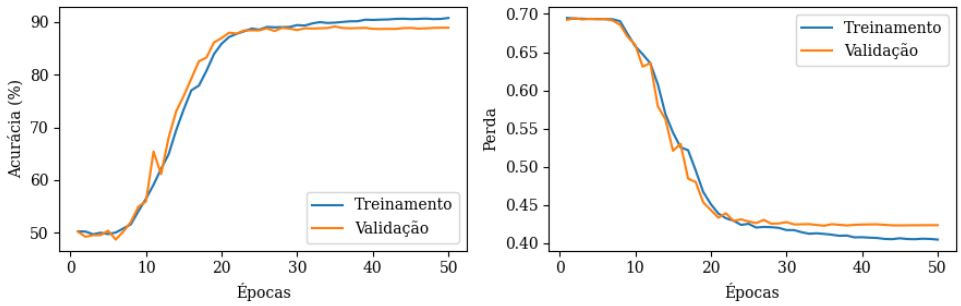
Figura 32 – Curvas de treinamento e teste referentes aos áudios de 5s.



(a) STFT.



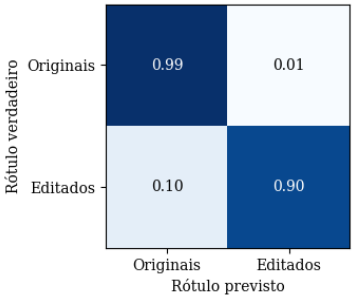
(b) Mel-STFT.



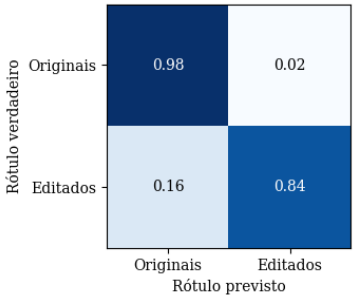
(c) CQT.

Fonte: Autoria própria.

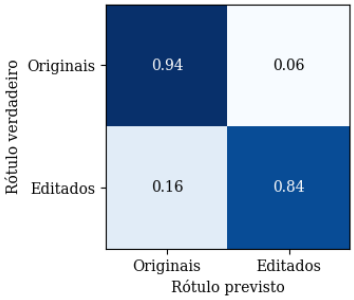
Figura 33 – Matrizes de confusão referentes aos áudios de 5s.



(a) STFT.



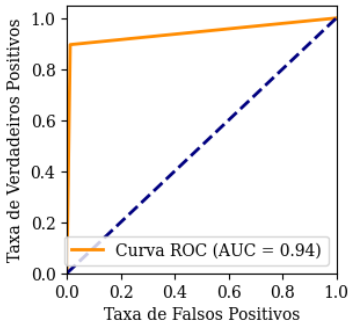
(b) Mel-STFT.



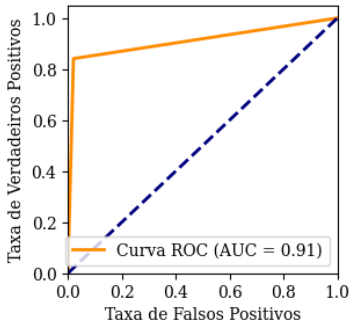
(c) CQT.

Fonte: Autoria própria.

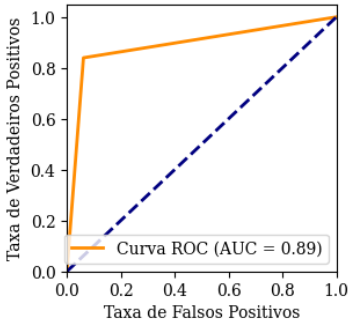
Figura 34 – Curvas ROC referentes aos áudios de 5s.



(a) STFT.



(b) Mel-STFT.



(c) CQT.

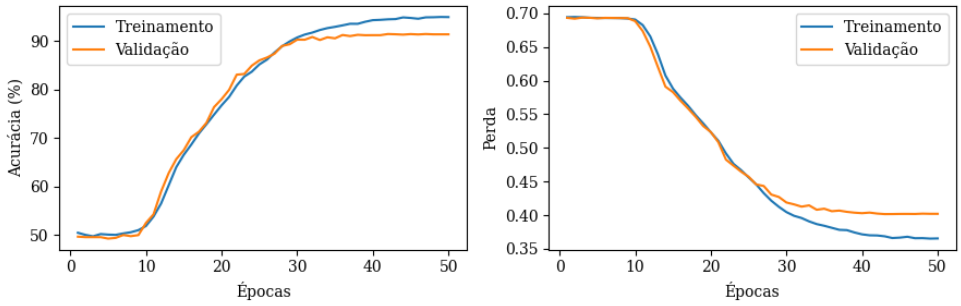
Fonte: Autoria própria.

A Figura 35 apresenta as curvas de acurácia média e perda média ao longo das cinco iterações para os conjuntos de treinamento e teste dos áudios de 6 segundos de duração.

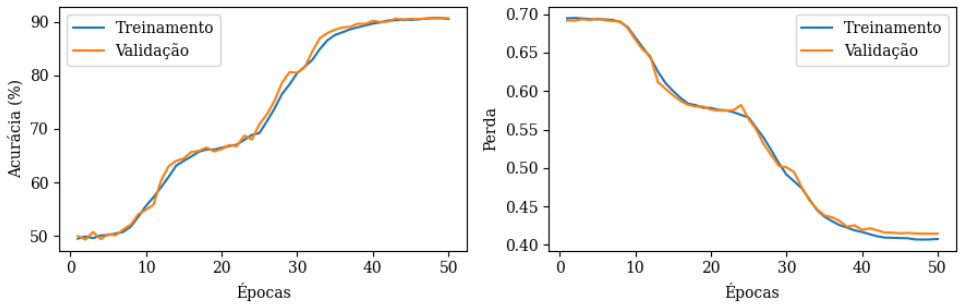
Na Figura 36 são ilustradas as matrizes de confusão normalizadas após as cinco iterações para os conjuntos de teste dos áudios de 6 segundos de duração.

A curva ROC é visualizada na Figura 37 após as cinco iterações para os conjuntos de teste dos áudios de 6 segundos de duração.

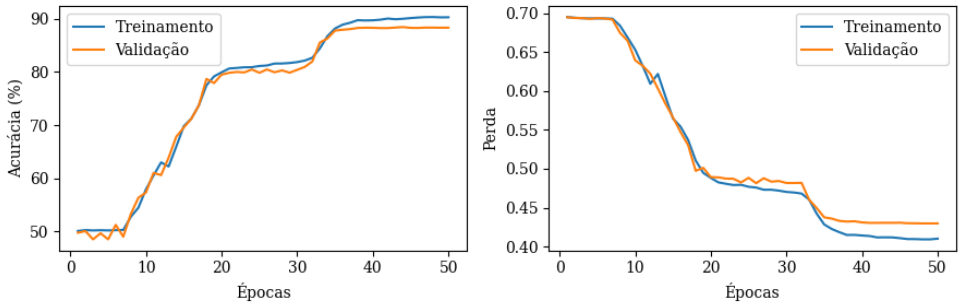
Figura 35 – Curvas de treinamento e teste referentes aos áudios de 6s.



(a) STFT.



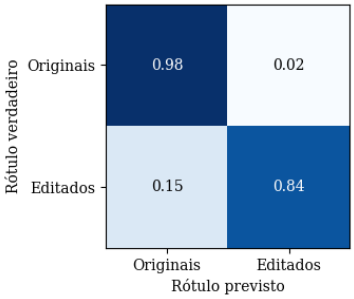
(b) Mel-STFT.



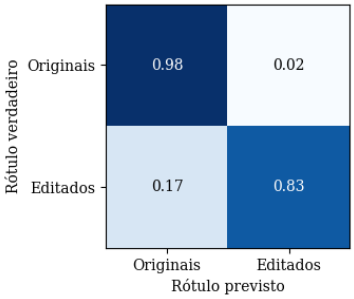
(c) CQT.

Fonte: Autoria própria.

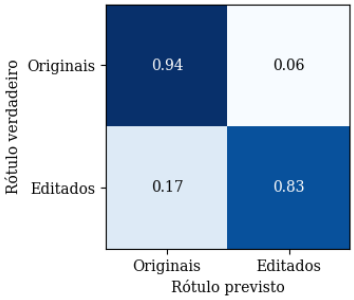
Figura 36 – Matrizes de confusão referentes aos áudios de 6s.



(a) STFT.



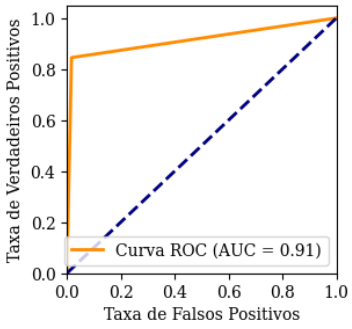
(b) Mel-STFT.



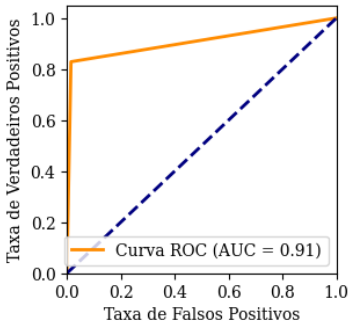
(c) CQT.

Fonte: Autoria própria.

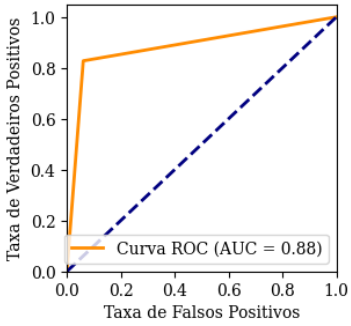
Figura 37 – Curvas ROC referentes aos áudios de 6s.



(a) STFT.



(b) Mel-STFT.



(c) CQT.

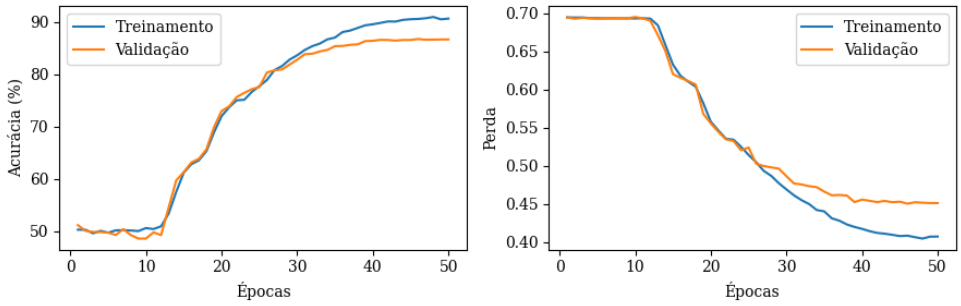
Fonte: Autoria própria.

A Figura 38 apresenta as curvas de acurácia média e perda média ao longo das cinco iterações para os conjuntos de treinamento e teste dos áudios de 7 segundos de duração.

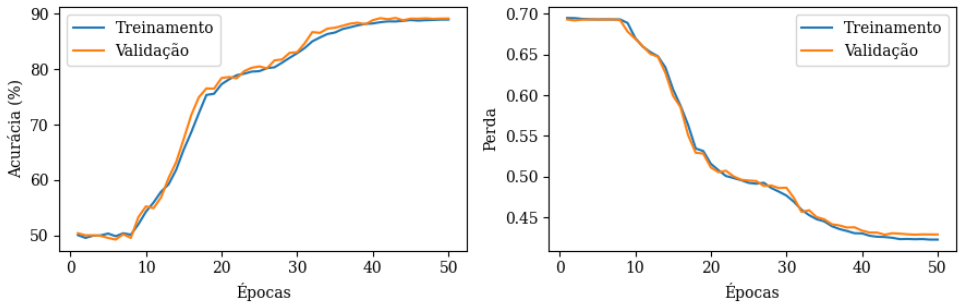
Na Figura 39 são ilustradas as matrizes de confusão normalizadas após as cinco iterações para os conjuntos de teste dos áudios de 7 segundos de duração.

A curva ROC é visualizada na Figura 40 após as cinco iterações para os conjuntos de teste dos áudios de 7 segundos de duração.

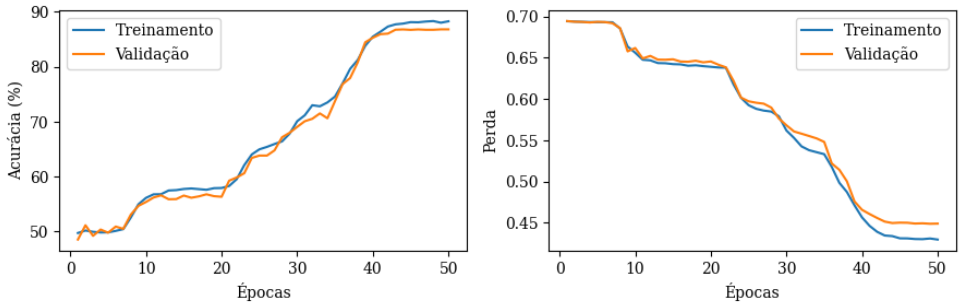
Figura 38 – Curvas de treinamento e teste referentes aos áudios de 7s.



(a) STFT.



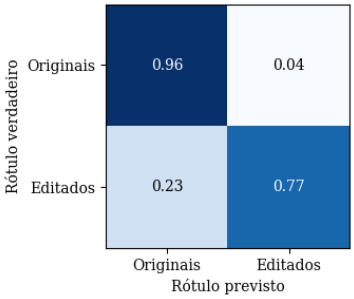
(b) Mel-STFT.



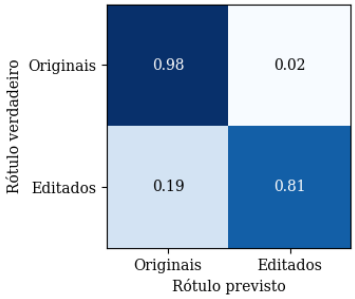
(c) CQT.

Fonte: Autoria própria.

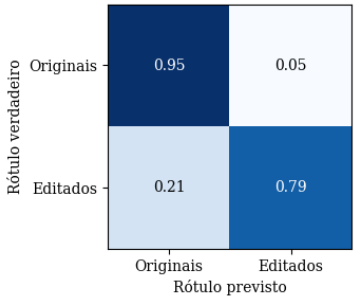
Figura 39 – Matrizes de confusão referentes aos áudios de 7s.



(a) STFT.



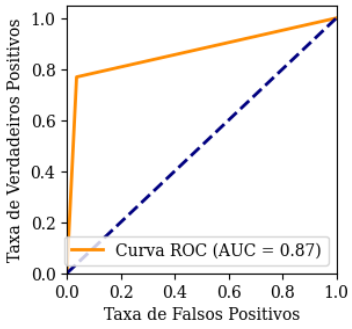
(b) Mel-STFT.



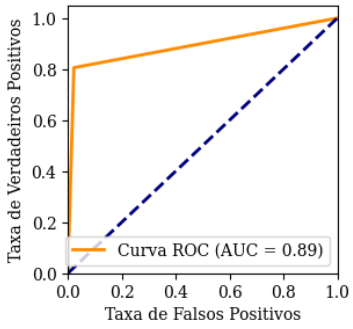
(c) CQT.

Fonte: Autoria própria.

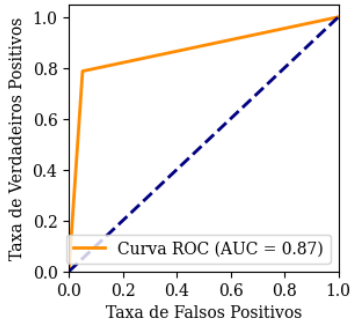
Figura 40 – Curvas ROC referentes aos áudios de 7s.



(a) STFT.



(b) Mel-STFT.



(c) CQT.

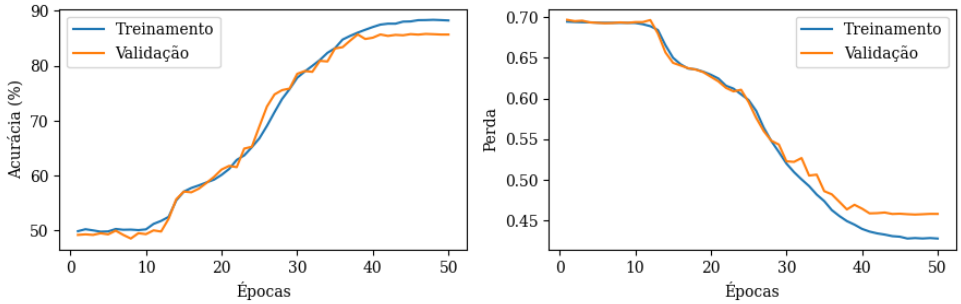
Fonte: Autoria própria.

A Figura 41 apresenta as curvas de acurácia média e perda média ao longo das cinco iterações para os conjuntos de treinamento e teste dos áudios de 8 segundos de duração.

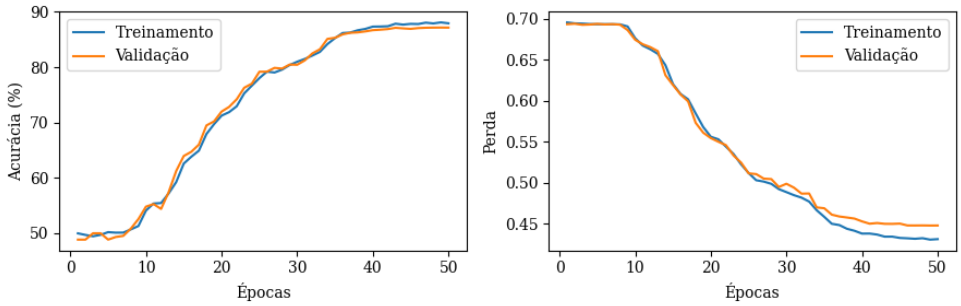
Na Figura 42 são ilustradas as matrizes de confusão normalizadas após as cinco iterações para os conjuntos de teste dos áudios de 8 segundos de duração.

A curva ROC é visualizada na Figura 43 após as cinco iterações para os conjuntos de teste dos áudios de 8 segundos de duração.

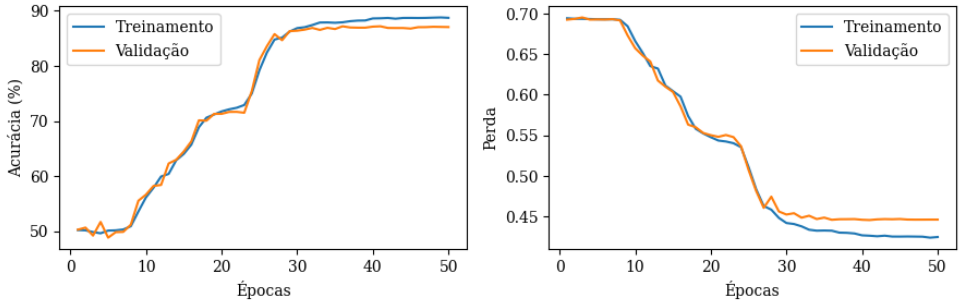
Figura 41 – Curvas de treinamento e teste referentes aos áudios de 8s.



(a) STFT.



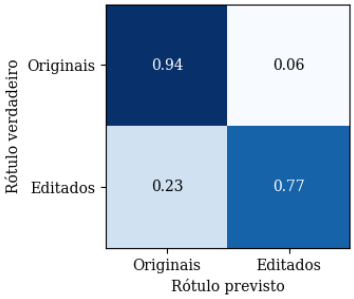
(b) Mel-STFT.



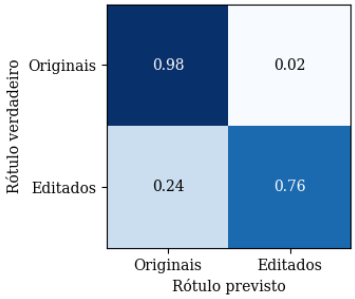
(c) CQT.

Fonte: Autoria própria.

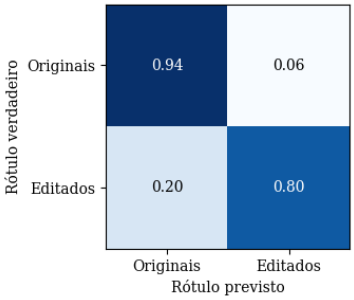
Figura 42 – Matrizes de confusão referentes aos áudios de 8s.



(a) STFT.



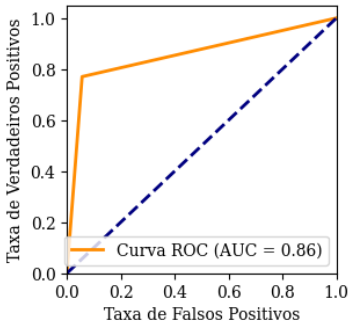
(b) Mel-STFT.



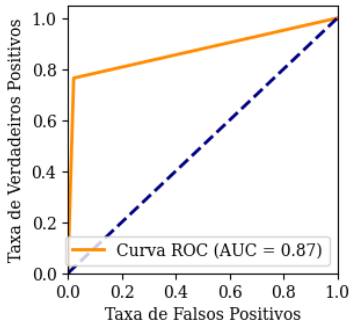
(c) CQT.

Fonte: Autoria própria.

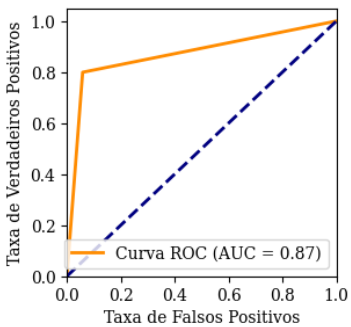
Figura 43 – Curvas ROC referentes aos áudios de 8s.



(a) STFT.



(b) Mel-STFT.



(c) CQT.

Fonte: Autoria própria.

A Tabela 2 apresenta os resultados obtidos para a detecção de edições no conjunto de teste para as diferentes representações de análise tempo-frequência em função da variação da duração dos áudios originais em que os segmentos foram inseridos.

	Linear-STFT	Mel-STFT	CQT
4s	95.28 ± 0.58	95.04 ± 2.15	90.19 ± 0.60
5s	94.17 ± 1.35	91.01 ± 2.40	88.90 ± 1.03
6s	91.39 ± 5.10	90.67 ± 4.55	88.34 ± 0.58
7s	86.65 ± 7.54	89.16 ± 1.34	86.81 ± 0.99
8s	85.69 ± 7.28	87.16 ± 2.73	87.04 ± 0.85

Tabela 2 – Acurácia (%) média e desvio-padrão da rede em classificar corretamente os áudios originais e editados em função da variação da duração dos áudios originais e da representação tempo-frequência.

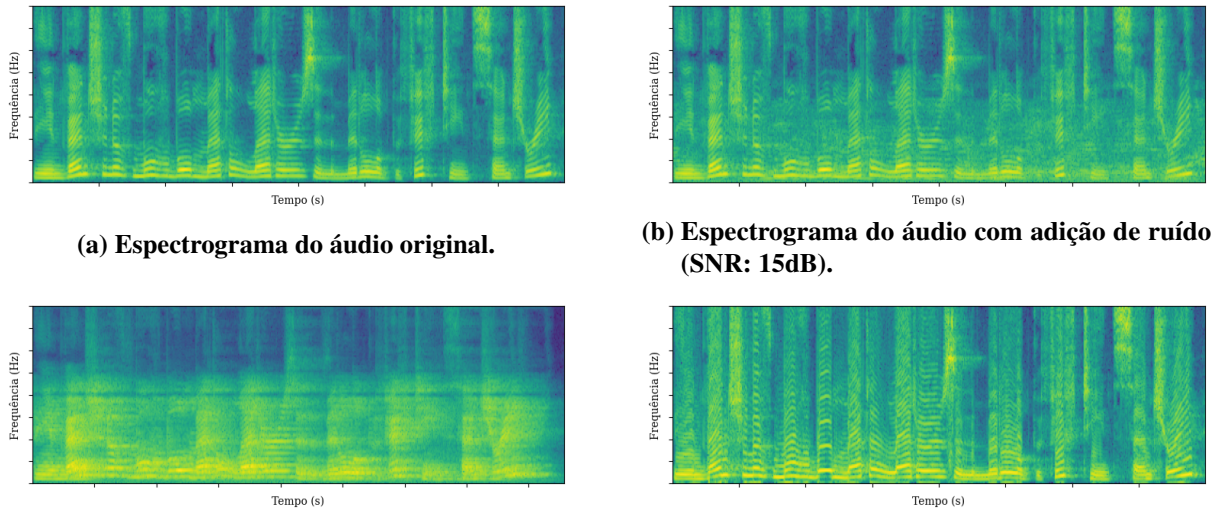
Para efeito de avaliação de desempenho, foi utilizado como linha de base (*baseline*) um modelo rede neural que não inclui as camadas convolucionais, constituído somente das camadas totalmente conectadas descritas no Capítulo 4. Os espectrogramas da STFT linear dos áudios de 4s foram convertidos em vetores unidimensionais para o processamento na rede. O resultado para a acurácia no conjunto de teste foi de 62.85 ± 1.35 %. Também foram feitas alterações no número de unidades de saída da primeira camada, não se obtendo melhorias nos resultados. Um experimento adicional foi a utilização direta dos áudios de 4s como dado de entrada das camadas totalmente conectadas, obtendo-se resultados de classificação totalmente aleatórios.

Diversas operações de pós-processamento de áudio podem ser realizadas com o intuito de mascarar uma edição e dificultar a sua detecção. A fim de testar a robustez do método desenvolvido, foram realizadas as seguintes operações nos sinais de áudio: adição de sinal ruidoso com razão sinal-ruído (SNR - *signal-to-noise ratio*) de 15 e 10 dB, simulação de reverberação através da convolução dos áudios com um sinal de amostra de resposta ao impulso de um ambiente, e utilização de filtro equalizador de pico com frequência central de 1kHz, ganho de 6 dB na frequência central e fator $Q = 0{,}7$.

Os efeitos das diferentes operações são ilustrados na Figura 44. A Figura 44(a) apresenta o espectrograma de um áudio original e as demais apresentam o espectrograma do áudio após o pós-processamento correspondente.

A Tabela 3 apresenta os resultados obtidos para a detecção de edições no conjunto de teste de áudios de 5 segundos de duração para as diferentes representações de análise tempo-frequência após a aplicação de diferentes operações de pós-processamento.

Figura 44 – Espectrogramas de um áudio após diferentes operações de pós-processamento.



(a) Espectrograma do áudio original.

(b) Espectrograma do áudio com adição de ruído (SNR: 15dB).

(c) Espectrograma do áudio com simulação de reverberação.

(d) Espectrograma do áudio com filtro de equalização.

Fonte: Autoria própria.

	Linear-STFT	Mel-STFT	CQT
Sem efeito	94.17 ± 1.35	91.01 ± 2.40	88.90 ± 1.03
Ruído (SNR:15dB)	79.76 ± 4.92	88.56 ± 4.96	87.09 ± 1.08
Ruído (SNR:10dB)	79.30 ± 2.25	88.65 ± 4.42	84.22 ± 1.50
Reverberação	89.59 ± 0.57	87.80 ± 1.12	88.53 ± 0.85
Filtro EQ	76.25 ± 11.50	81.04 ± 3.45	77.53 ± 0.55

Tabela 3 – Acurácia (%) média e desvio-padrão da rede em classificar corretamente os áudios de 5 segundos originais e editados em função da aplicação de operações de pós-processamento e da representação tempo-frequência.

5.2 RESULTADOS DE TRABALHOS CORRELATOS

Uma vez que não existem *datasets* públicos com um número considerável de amostras, os trabalhos anteriores que trataram do tema efetuaram o procedimento de geração dos áudios editados previamente ao desenvolvimento dos modelos de detecção. A tarefa de comparação dos resultados de diferentes trabalhos é dificultada pelo fato de que os *datasets* e os procedimentos utilizados para a geração da base de dados são diferentes. A acurácia de detecção do modelo depende fundamentalmente do modo como os áudios são editados, e boa parte dos trabalhos não detalha o procedimento de edição adotado. A seguir são apresentados os resultados dos trabalhos descritos no Cap. 3.

Shi e Ma (2011) efetuaram a gravação dos áudios utilizados nos experimentos com diferentes taxas de amostragem (6, 8, 11.025, 16, 22.025, 32, 44.1, 48 e 96 kHz), sendo gerados 60 áudios igualmente divididos entre originais e editados para cada taxa. A interpolação foi feita com a inserção de áudios em posições aleatórias. Algumas amostras foram editadas com o

software Cool Edit e outras com o MATLAB. Os áudios possuem 8 palavras e as edições foram realizadas com diferentes fatores de interpolação. As acurácias informadas apresentam valores de 80 a 100%.

No estudo de Pan *et al.* (2012), os autores coletaram 452 áudios aleatórios de duração aproximada de 3s do *dataset* TIMIT. Os sinais foram processados através da adição de ruído com diferentes SNRs: 10, 15, 20 e 30 dB. Para cada nível de ruído foram gerados 100 áudios editados. Foram apresentadas as curvas ROC para os diferentes níveis de ruido com o valor da área sob a curva diminuindo conforme o aumento do SNR.

O *dataset* TIMIT e um *dataset* não identificado com 1000 amostras gravadas com quatro microfones comercias em diferentes ambientes foram utilizados nos experimentos Zhao *et al.* (2014). Foi realizada a simulação de reverberação nos sinais e realizada a adição de ruido. Os segmentos foram inseridos em posições aleatórias nos áudios. Foram reportadas diversas acurácias entre 100 e 50%, dependendo do tipo de pós-processamento efetuado, do *dataset* e das *features* utilizadas.

O trabalho de Hua *et al.* (2016) utilizou dez gravações de áudio com diferentes gravadores em ambientes diversos, com durações que variaram entre 4 e 20 minutos. Os segmentos inseridos possuem a proporção de aproximadamente um sexto da extensão total da gravação. Foram avaliados dois algoritmos, com diferentes taxas de detecção de erros.

Capoferri *et al.* (2020) utilizou o *dataset* ACE para a criação de um conjunto de 20000 áudios, igualmente dividido entre originais e editados com operações aleatórias de interpolação. Foram efetuadas operações de simulação de reverberação e adições de ruido. O estudo apresentou as curvas ROC para diferentes simulações com resultados de alta variabilidade (AUC entre 1 e 0,5).

A acurácia da rede desenvolvida no trabalho de Jadhav *et al.* (2019) variou entre 70,87% e 98,31% para a detecção de edição em uma base dados de 4000 áudios originais e 4400 áudios editados a partir da base de dados Free Spoken Digit Database (FDSS). O *dataset* contém áudios de quatro falantes com diferentes *pitches* (percepção da frequência fundamental) pronunciando dígitos de 0 a 9. No trabalho, dígitos de um determinado falante foram inseridos no meio de áudios de outro. Foram efetuados testes com a inserção de um a três dígitos e os resultados variaram consideravelmente.

Ustubioglu *et al.* (2022) obtiveram uma acurácia de 95% e 99% (com *data augmentation*) para a detecção de réplicas em uma base de dados com 368 áudios editados a partir do

dataset TIMIT e 1329 áudios do *dataset* Arabic Corpus Speech. Os áudios originais contém durações que variam de 2 a 6 segundos e os segmentos repetidos inseridos inseridos em posições aleatórias nos áudios variou entre 0,2 e 0,6 segundos.

O trabalho de Moussa *et al.* (2022) reportou as melhores acurácias com valores acima de 90% para a detecção de áudios interpolados com a utilização dos *datasets* ACE e Hi-FI TTS. O processo de geração dos áudios adulterados inclui múltiplas combinações entre áudios de um mesmo falante e a convolução dos áudios com um sinal de resposta ao impulso de um ambiente (RIR) para simulação de reverberação.

As acurácias reportadas no trabalho de Zeng e Wu (2022) estão na faixa entre 42% e 90,3%, com variação da duração do segmento inserido como entrada para o modelo. O *dataset* LibriSpeech corpus foi utilizado para os procedimentos de edição, com a utilização de regras para as operações de exclusão, substituição de trechos dos áudios.

Apesar de ser difícil realizar a comparação, considerando os diversos motivos elencados, observa-se que o presente trabalho utilizou uma base de dados com um número consideravelmente maior de amostras que a maioria dos trabalhos descritos, além de procedimentos mais complexos para a geração dos arquivos adulterados. Alguns dos estudos utilizam operações aleatórias básicas para a interpolação dos áudios, enquanto outros nem mesmo descrevem ou detalham o procedimento efetuado. O percentual do trecho editado dos trabalhos também não foi especificado ou descrito com clareza. Ainda, alguns métodos, apesar de utilizarem frases de um mesmo falante, adicionam conteúdo espectral diferente nos diversos arquivos.

6 CONCLUSÃO

Foi proposto o desenvolvimento de um método baseado em aprendizagem profunda para detecção de edições em registros de áudio. O modelo empregado foi capaz de extrair características dos espectrogramas processados e realizar com êxito a tarefa de classificação entre áudios originais e editados, mesmo após a condução de diversas operações de pós-processamento nos registros questionados. Também foram efetuados diversos procedimentos a fim de garantir que, apesar do fato de que as locuções inseridas no trecho original fossem emitidas por um falante distinto, as edições se aproximassem de uma situação real. A utilização de representações de análise tempo-frequência, como espectrograma, mel-espectrograma e espectrograma da transformada Q constante, permitiu ao modelo capturar padrões acústicos e identificar as diferenças entre áudios não editados e aqueles que sofreram adulterações.

Conforme esperado, foi constatada a tendência de diminuição da acurácia conforme o aumento da duração dos áudios originais, tendo em vista a diminuição da proporção do trecho editado em relação à extensão total do sinal de áudio. A redução da adulteração implica em uma menor dissimilaridade de conteúdo espectral entre os gráficos pertencentes às duas classes.

Ao analisar os resultados obtidos, é possível observar que a rede obteve um desempenho superior com a entrada da representação obtida através da STFT linear em relação às demais para os áudios de duração de até 6s. A acurácia média foi similar para os áudios de 8s, entretanto, a diminuição do número de acertos de classificação foi consideravelmente menor para a CQT conforme o aumento da duração dos áudios originais. O desvio-padrão também apresenta um valor menor para a CQT, indicando uma maior consistência nos resultados. O mel-espectrograma apresentou um desempenho melhor para os áudios de 7s e os resultados foram mais inconsistentes com a utilização do espectrograma em escala de frequências dispostas linearmente para os áudios de duração maior.

Em todos os cenários analisados, a acurácia de detecção de áudios originais foi consideravelmente superior em relação aos editados, com valores próximos de 100% no processamento dos áudios de menor duração e 94 a 96% para os áudios de 8s. É interessante notar que a CQT obteve o melhor desempenho (80%) para a identificação dos áudios adulterados de 8s, quando comparada com as representações da STFT (76 e 77%).

Uma limitação do modelo é a impossibilidade de detecção de áudios editados quando não existe diferença significativa no padrão de ruído entre o trecho original e o segmento inserido.

Conforme exposto no Cap. 4, foram realizadas tentativas de detecção somente com a utilização do *dataset* LJSpeech, não se obtendo êxito, uma vez que os áudios foram gerados através de segmentação da gravação original. Portanto, o método desenvolvido não foi eficaz para tratar do problema de edição do tipo réplica.

Trabalhos futuros poderiam incluir a utilização de diferentes representações tempo-frequência, como, por exemplo, a transformada *wavelet*, transformada Gabor e a distribuição de Wigner-Ville. Também poderiam ser desenvolvidas arquiteturas de redes neurais convolucionais mais complexas e com quantidades maiores de parâmetros, ou utilizadas redes amplamente reconhecidas, como a AlexNet, VGGNet, ResNet, MobileNet, etc. Sugere-se ainda o emprego de procedimentos mais elaborados para a geração de áudios editados, que poderiam incluir uma etapa de transcrição dos áudios para a substituição das locuções. Por fim, os testes podem ser realizados em diferentes bases de dados, tais como: TIMIT, CommmonVoice, LibriSpeech, TED-LIUM, WSJ, dentre outras.

Diante do exposto, fica evidenciado o potencial da aprendizagem profunda para impulsionar avanços contínuos e abrir novas perspectivas para o aperfeiçoamento dos métodos de análise na área de áudios forenses.

REFERÊNCIAS

BROWN, Judith C. Calculation of a constant q spectral transform. **The Journal of the Acoustical Society of America**, Acoustical Society of America, v. 89, n. 1, p. 425–434, 1991.

CAPOFERRI, Davide; BORRELLI, Clara; BESTAGINI, Paolo; ANTONACCI, Fabio; SARTI, Augusto; TUBARO, Stefano. Speech audio splicing detection and localization exploiting reverberation cues. *In*: IEEE. **2020 IEEE International Workshop on Information Forensics and Security (WIFS)**. *[S.l.]*, 2020. p. 1–6.

CHOLLET, Francois. **Deep learning with Python**. *[S.l.]*: Simon and Schuster, 2021.

COOLEY, James W; TUKEY, John W. An algorithm for the machine calculation of complex fourier series. **Mathematics of computation**, v. 19, n. 90, p. 297–301, 1965.

DINIZ, Paulo SR; SILVA, Eduardo AB da; NETTO, Sergio L. **Processamento digital de sinais-: Projeto e análise de sistemas**. *[S.l.]*: Bookman Editora, 2014.

GÉRON, Aurélien. **Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow**. *[S.l.]*: "O'Reilly Media, Inc.", 2022.

HAYKIN, Simon S; VEEN, Barry Van. **Sinais e sistemas**. *[S.l.]*: Bookman, 2001.

HOUCK, Max M; SIEGEL, Jay A. **Fundamentals of forensic science**. *[S.l.]*: Academic Press, 2009.

HUA, Guang; ZHANG, Ying; GOH, Jonathan; THING, Vrizlynn LL. Audio authentication by exploring the absolute-error-map of enf signals. **IEEE Transactions on Information Forensics and Security**, IEEE, v. 11, n. 5, p. 1003–1016, 2016.

INMAN, Keith; RUDIN, Norah. **Principles and practice of criminalistics: the profession of forensic science**. *[S.l.]*: CRC Press, 2000.

ITO, Keith; JOHNSON, Linda. **The LJ Speech Dataset**. 2017. https://keithito.com/LJ-Speech-Dataset/.

JADHAV, Shital; PATOLE, Rashmika; REGE, Priti. Audio splicing detection using convolutional neural network. *In*: IEEE. **2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)**. *[S.l.]*, 2019. p. 1–5.

KAMATH, Uday; LIU, John; WHITAKER, James. **Deep learning for NLP and speech recognition**. *[S.l.]*: Springer, 2019. v. 84.

LATHI, Bhagwandas Pannalal. **Sinais e sistemas lineares-2**. *[S.l.]*: Bookman, 2006.

LIDY, Thomas; SCHINDLER, Alexander. Cqt-based convolutional neural networks for audio scene classification. *In*: **DCASE**. *[S.l.: s.n.]*, 2016. p. 60–64.

LOSHCHILOV, Ilya; HUTTER, Frank. Decoupled weight decay regularization. **arXiv preprint arXiv:1711.05101**, 2017.

MAHER, Robert C. Overview of audio forensics. **Intelligent multimedia analysis for security applications**, Springer, p. 127–144, 2010.

MCLOUGHLIN, Ian; XIE, Zhipeng; SONG, Yan; PHAN, Huy; PALANIAPPAN, Ramaswamy. Time–frequency feature fusion for noise robust audio event classification. **Circuits, Systems, and Signal Processing**, Springer, v. 39, n. 3, p. 1672–1687, 2020.

MOUSSA, Denise; HIRSCH, Germans; RIESS, Christian. Towards unconstrained audio splicing detection and localization with neural networks. **arXiv preprint arXiv:2207.14682**, 2022.

MÜLLER, Meinard. **Fundamentals of music processing: Audio, analysis, algorithms, applications**. *[S.l.]*: Springer, 2015. v. 5.

PAN, Xunyu; ZHANG, Xing; LYU, Siwei. Detecting splicing in digital audios using local noise level estimation. *In*: IEEE. **2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. *[S.l.]*, 2012. p. 1841–1844.

RASCHKA, Sebastian. **Python machine learning**. *[S.l.]*: Packt publishing ltd, 2015.

SHI, Qian; MA, Xiaohong. Detection of audio interpolation based on singular value decomposition. *In*: IEEE. **2011 3rd International Conference on Awareness Science and Technology (iCAST)**. *[S.l.]*, 2011. p. 287–290.

STEVENS, Eli; ANTIGA, Luca; VIEHMANN, Thomas. **Deep learning with PyTorch**. *[S.l.]*: Manning Publications, 2020.

USTUBIOGLU, Arda; USTUBIOGLU, Beste; ULUTAS, Guzin. Mel spectrogram-based audio forgery detection using cnn. **Signal, Image and Video Processing**, Springer, p. 1–9, 2022.

WARDEN, Pete. Speech commands: A dataset for limited-vocabulary speech recognition. **arXiv preprint arXiv:1804.03209**, 2018.

ZAKARIAH, Mohammed; KHAN, Muhammad Khurram; MALIK, Hafiz. Digital multimedia audio forensics: past, present and future. **Multimedia tools and applications**, Springer, v. 77, p. 1009–1040, 2018.

ZENG, Zhiping; WU, Zhizheng. Audio splicing localization: Can we accurately locate the splicing tampering? *In*: IEEE. **2022 13th International Symposium on Chinese Spoken Language Processing (ISCSLP)**. *[S.l.]*, 2022. p. 120–124.

ZHAO, Hong; CHEN, Yifan; WANG, Rui; MALIK, Hafiz. Audio source authentication and splicing detection using acoustic environmental signature. *In*: **Proceedings of the 2nd ACM workshop on Information hiding and multimedia security**. *[S.l.: s.n.]*, 2014. p. 159–164.

ZIABARY, Pedram Abdzadeh; VEISI, Hadi. A countermeasure based on cqt spectrogram for deepfake speech detection. *In*: IEEE. **2021 7th International Conference on Signal Processing and Intelligent Systems (ICSPIS)**. *[S.l.]*, 2021. p. 1–5.

ZJALIC, James. **Digital audio forensics fundamentals: from capture to courtroom**. *[S.l.]*: CRC Press, 2020.