UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

THALLIA CAVEION

CLASSIFICAÇÃO DE SEGMENTOS DE VÍDEOS POR TRECHOS DE FALA UTILIZANDO REDE NEURAL CONVOLUCIONAL

PATO BRANCO

2022

THALLIA CAVEION

CLASSIFICAÇÃO DE SEGMENTOS DE VÍDEOS POR TRECHOS DE FALA UTILIZANDO REDE NEURAL CONVOLUCIONAL

Classification of video segments by speech snippets using convolutional neural network

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharela em Engenharia de Computação da Universidade Tecnológica Federal do Paraná (UTFPR).

Orientador: Prof. Dr. Érick Oliveira Rodrigues
Coorientador: Prof. Dr. Jefferson Tales Oliva

PATO BRANCO

2022

THALLIA CAVEION

CLASSIFICAÇÃO DE SEGMENTOS DE VÍDEOS POR TRECHOS DE FALA UTILIZANDO REDE NEURAL CONVOLUCIONAL

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharela em Engenharia de Computação da Universidade Tecnológica Federal do Paraná (UTFPR).

Data de Aprovação: 05 de dezembro de 2022.

Prof. Dr. Érick Oliveira Rodrigues
Universidade Tecnológica Federal do Paraná

Prof. Dr. Jefferson Tales Oliva
Universidade Tecnológica Federal do Paraná

Prof. Dr. Dalcimar Casanova
Universidade Tecnológica Federal do Paraná

Profa. Dra. Rubia Eliza De Oliveira Schultz Ascari
Universidade Tecnológica Federal do Paraná

PATO BRANCO

2022

RESUMO

CAVEION, Thallia. Classificação de segmentos de vídeos por trechos de fala utilizando rede neural convolucional. 2022. 45 f. Trabalho de Conclusão de Curso – Curso de Engenharia de Computação, Universidade Tecnológica Federal do Paraná. Pato Branco, 2022.

Vídeos são uma das formas mais populares de conteúdo na Internet, Plataformas de mídia social e telefones celulares tornaram simples e rápido capturar e publicar vídeos. No entanto, em diversos casos se faz necessário a edição manual desse tipo de conteúdo, a qual é considerada custosa, pois demanda muito tempo. Portanto, a necessidade de uma alternativa de edição automática. Neste trabalho, foi proposto um modelo de classificação de trechos de vídeo utilizando rede neural convolucional. A entrada é composta por diversos vídeos de autoria própria que foram cortados de forma aleatória em diversos trechos. Destes trechos foram extraídos os áudios obtendo arquivos no formato wav (waveform audio file format) que foram previamente classificados. Posteriormente, são extraídas características de cada arquivo utilizando o método coeficientes cepstrais de frequência de Mel (MFCC), obtendo assim espectrogramas equivalentes a imagens 2D. Estes espectrogramas servem como entrada para a rede neural convolucional ao qual gera como saída um modelo de classificação. Resultados experimentais da metodologia proposta mostraram que o modelo é capaz de classificar 92,52% dos trechos de forma correta.

Palavras-chave: Rede neural convolucional; Classificação de trechos de vídeo; Coeficientes cepstrais de frequência de Mel; Edição de vídeo.

ABSTRACT

CAVEION, Thallia. Classification of video segments by speech snippets using convolutional neural network. 2022. 45 f. Completion of course work – Computer Engineering Course, Federal Technological University of Paraná. Pato Branco, 2022.

Videos are one of the most popular contents on the internet, social medias platforms and cell phone's made capture and publish videos so easy and fast. However, in a lot of cases it's necessary edit the content manually, which is irksome, because it takes a lot of time. Therefore, the need automatic editing alternative. In this final paper presents a creation proposal of video classification model, using convolutional neural networks. Starts with several self-authored videos, which each one was randomly cut in different parts. From these parts, there were took the audios in wav (waveform audio format), previously classified and after extracted the characteristics of each file using the Mel frequency cepstrum coefficients method (MFCC), it was got spectrograms as like 2D images. Those spectrograms start the convolutional neural networks implemented producing as output, a classification model. Experimental results from the methodology proposed, showed that this model it's able to classify 92,52% of videos stretches correctly.

Keywords: Convolutional neural network; Classification of video clips; Coefficients Mel's frequency cepstrals; Video editing.

LISTA DE CÓDIGOS-FONTE

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

SIGLAS

AM	Aprendizado de Máquina
CMN	*Cepstral Mean Normalization*
CNN	*Convolution Neural Network*
DCT	*Discrete Cosine Transform*
FFT	*Fast Fourier Transform*
HMM	*Hidden Markov Model*
IA	Inteligência Artificial
MFCC	*Mel- Frequency Cepstral Coefficients*
RNAs	*Artificial Neural Networks*
STFT	*Short Time Fourier Transform*

SUMÁRIO

1 INTRODUÇÃO

Vídeos são uma das formas mais populares de conteúdo na Internet. De acordo com uma projeção realizada pela Cisco (2017), esse tipo de conteúdo será responsável por cerca de 80% de todo o tráfego da Internet até 2021. Plataformas de mídia social e telefones celulares tornaram mais fácil e rápido capturar e publicar vídeos. No entanto, a edição manual desses vídeos é ainda considerada custosa, pois demanda muito tempo.

De acordo com Okun, Susan Zwerman *et al.* (2015), a edição de vídeo é definida como o ato de cortar e juntar pedaços de uma ou mais fontes para a criação de conteúdo. As ferramentas de edição de vídeo podem ser definidas como programas de computador em que os usuários podem fazer essa tarefa, ou seja, combinar segmentos de vídeo.

Muitas vezes, em um vídeo podem haver trechos a serem descartados, por exemplo, momentos onde não há fala de pessoas. A detecção destes trechos ocorre no momento da edição manual onde o editor assiste o vídeo em questão, realiza os cortes necessários e descarta determinados trechos.

Atualmente existem algumas iniciativas de automatização para facilitar o processamento de vídeos, como o *Silver* (CASARES *et al.*, 2002) e o *Roughcut* (LEAKE *et al.*, 2017). Apesar dessa atividade de automatização de processamento de vídeos ser amplamente abordada dentro da área de inteligência artificial (IA) e Aprendizado de Máquina (AM), não existem muitos métodos ou ferramentas disponíveis que oferecem uma edição de vídeos de forma totalmente automatizada.

1.0.1 Objetivo Geral

Construção de um modelo, utilizando rede neural convolucional, para a classificação de trechos de vídeo.

1.1 JUSTIFICATIVA

Mesmo com o avanço da tecnologia, a criação de mídia de vídeo, ainda é uma tarefa considerada difícil quando comparado com a mídia convencional baseada em textos ou fotos. Sendo assim, é necessário algum tipo de suporte para esses sistemas de várias tecnologias de processamento de informação que pode ajudar o usuário a editar este tipo de conteúdo.

Este trabalho propõe a classificação de trechos para a segmentação de vídeos, tendo em visto que segmentos classificados como sem fala são descartados do vídeo final. Assim, poupando o tempo dos usuários com este tipo de tarefa, deixando assim esse processo de edição bruta mais rápido e fácil, podendo se ater apenas a outros tipos de tarefas. Esta classificação está dividida em 6 classes das quais são: a classe 0 para trechos que não contém fala alguma; 1 para trechos de vídeo que contém apenas momentos de fala; 2, como trechos mistos onde pode conter tanto momentos sem fala quanto momentos com fala; 3, para trechos sem fala que contém ruídos de fundo; 4 para trechos com fala que contém ruídos; e 5, como trechos mistos com ruídos de fundo onde pode conter tanto momentos sem fala quanto momentos com fala.

1.2 ESTRUTURA DO TRABALHO

O trabalho está organizado da seguinte forma. A próxima seção contém uma revisão sobre o referencial teórico com temas ao qual serão utilizados para a classificação de trechos de vídeo, como a estruturação de um vídeo, características de som, métodos de classificação e trabalhos relacionados. A seção 3 apresenta a proposta principal desse trabalho, como é realizada a extração de características de áudio dos vídeos, métodos de classificação e técnicas utilizadas para o desenvolvimento. A seção 4 apresenta como foi realizada implementação para obtenção do modelo de classificação de trechos de vídeos. Na seção 5 à a conclusão do presente trabalho.

2 REFERENCIAL TEÓRICO

Neste capítulo são apresentados os fundamentos teóricos. São descritos os conceitos centrais relacionados à estruturação de vídeos, características de áudio, aos conceitos de Aprendizado de Máquina, noções de métodos de classificação que serão abordados no presente trabalho. Por fim, são apresentadas metodologias propostas em trabalhos relacionados.

2.1 ESTRUTURA DE VÍDEO

Os blogs podem ser descritos como sites baseados em jornais, que usam ferramentas de gerenciamento de conteúdo para que os autores postem conteúdos nos sites. Em geral compartilham pontos de vista, interesses e opiniões, formando assim, uma comunidade virtual. Vlogs são semelhantes a blogs, mas ao invés de usar textos para transmitir mensagens, são utilizados vídeos, fornecendo assim maior liberdade de expressão e interação mais direta com os expectadores. Sua crescente popularidade resultou no aumento na ocorrência de tarefas de edição de vídeos (WARMBRODT; SHENG; HALL, 2008).

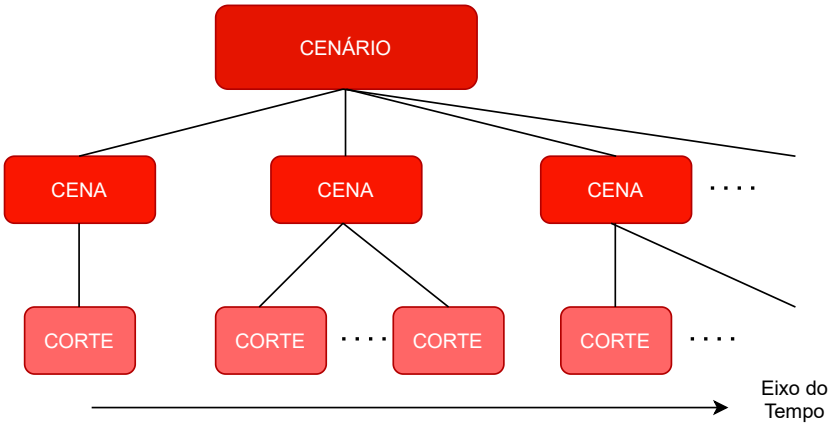
A edição de vídeo é o processo de reorganização ou modificação de segmentos de vídeos. Os objetivos da edição de vídeo são as mesmas da edição de filme: a remoção de filmagens indesejadas; o isolamento de clipes desejados; e o arranjo a tempo de sintetizar um novo pedaço de filmagem (GAO *et al.*, 2010).

A Figura 1 mostra um exemplo de estrutura hierárquica das informações do vídeo. O cenário é o vídeo inteiro composto por várias cenas. Cada cena consiste em um ou mais cortes. As cenas podem ter sub-cenas. Nesta estrutura, as partes superiores possuem as descrições, as partes inferiores têm descrições mais visuais. As partes mais baixas desta estrutura, ou seja, os cortes contêm matérias-primas do vídeo (UEDA *et al.*, 1993).

Outro cenário muito comum além de vlogs, são os vídeos de esportes, que contemplam uma grande audiência, considerados muito importantes em programas de televisão. Comparado a outros vídeos, têm estrutura de conteúdo e regras de domínio bem definidas. Os vídeos de esportes costumam ser divididos em alguns segmento, os quais também podem ter alguns sub-segmentos (ZHONG; CHANG, 2001). Por exemplo, um jogo de tênis é dividido primeiro em sets, depois em jogos e serviços como mostrado na Figura 2.

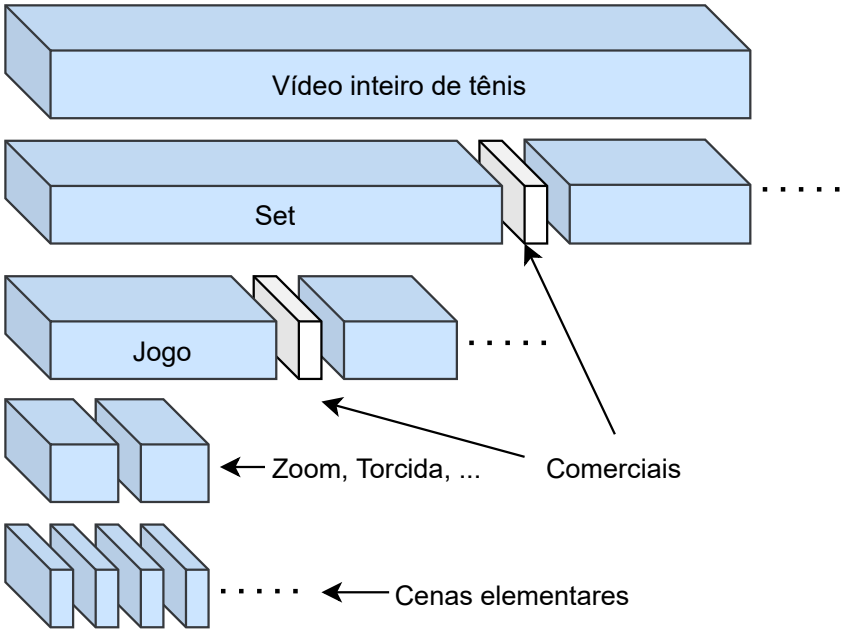
No tênis, quando um saque começa, a cena geralmente é mudada para a vista da quadra. Além disso, para transmissões em programas de TV, há comerciais ou outras informações, como

Figura 1 – Estrutura da informação do vídeo



Fonte: Adaptado de Ueda *et al.* (1993).

Figura 2 – Estrutura temporal típica de um vídeo programa de tênis



Fonte: Adaptado de Zhong e Chang (2001).

placar e nome do jogador inseridas entre as seções do jogo. As cenas elementares se caracterizam como cenas importantes do jogo que serão reapresentadas. como uma boa jogada realizada por algum jogador.

2.2 CARACTERÍSTICAS DE ÁUDIO

Um vídeo é composto por dois elementos principais, sendo eles imagem e áudio. O principal objetivo desse presente trabalho é a classificação de trechos de vídeo. Essa classificação foi realizada através de características de áudio, desta forma se torna importante a sua compreensão.

De acordo com Liu, Yao Wang e Tsuhan Chen (1998), a compreensão de áudio pode ser

dividida em três camadas: características acústicas de baixo nível; assinaturas de áudio de nível intermediário associadas a diferentes objetos sonoros; e modelos semânticos de alto nível de áudio em diferentes classes da cena.

Na camada de características acústicas, podem ser analisados recursos amplos de baixo nível, como volume, período de tom e largura de banda de um sinal de áudio. Na camada de assinatura acústica, pode-se determinar o objeto que produz um determinado som. Os sons que podem ser produzidos por diferentes objetos têm diferentes assinaturas. Ao armazenar essas determinadas assinaturas em um banco de dados e harmonizando com um segmento de áudio que deva ser classificado, é possível categorizar este segmento em uma classe de objeto. Na camada modelo de alto nível, podem ser regras semânticas (LIU; WANG, Y.; CHEN, T., 1998).

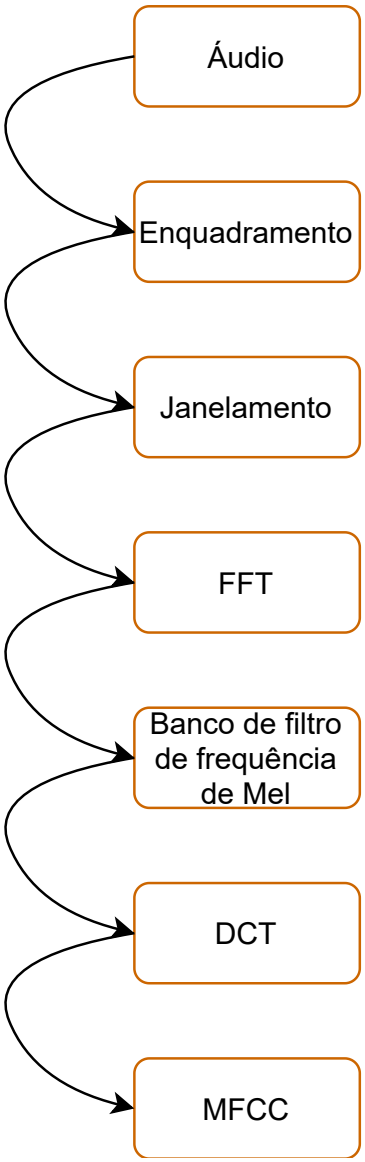
Existem muitas características que podem ser usadas para a caracterização de áudio. Geralmente, eles podem ser separados nas seguintes categorias: domínio de tempo, abordagem não linear, domínio de frequência e domínio de tempo-frequência (LIU; WANG, Y.; CHEN, T., 1998).

De acordo com Tun *et al.* (2020) spectrograma é a representação de um sinal de áudio, mostrando os espectros de frequência no decorrer do tempo, podem ser calculadas através da transformação rápida de Fourier, do inglês *Fast Fourier Transform* (FFT).

Para Tun *et al.* (2020), coeficientes cepstrais de frequência de Mel, do inglês *Mel-Frequency Cepstral Coefficients* (MFCC), é uma técnica de extração de características comumente utilizada. As características do MFCC podem ser convertidas em estatísticas para uso em tarefas de classificação. O processo de MFCC é composto pelas seguintes etapas: enquadramento, janelamento, transformada rápida de Fourier, banco de filtros de frequência de Mel e transformada discreta de cosseno. O processo do MFCC para a extração de características é mostrado na Figura 3

No enquadramento, o sinal de entrada de voz é segmentada em quadros (*frames*), que são multiplicados sob o auxílio da função janela deslizante de *Hamming* para suavizar o sinal. A transformada rápida de Fourier converte o sinal no domínio do tempo para o domínio da frequência. Sua entrada é o sinal janela e sua saída são as bandas de frequência discretas. A transformação discreta do cosseno, do inglês *Discrete Cosine Transform* (DCT), é processada na conversão do espectro de log Mel no domínio do tempo (TUN *et al.*, 2020).

Figura 3 – Etapas MFCC



Fonte: Adaptado de Tun *et al.* (2020).

2.3 APRENDIZADO DE MÁQUINA

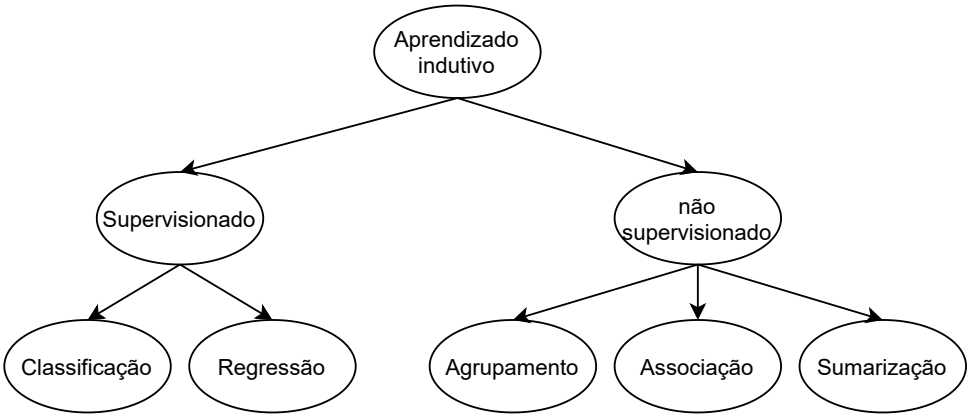
O Aprendizado de Máquina (AM) é uma das áreas emergentes na ciência da computação. Métodos de AM são desenvolvidos para a detecção automática de padrões significativos em dados. Com a quantidade de dados disponível cada vez maior, a análise manual dos mesmos se torna inviável (OSISANWO *et al.*, 2017).

De acordo com Faceli *et al.* (2011), algoritmos de AM podem ser utilizados em diversas tarefas, como preditivas e descritivas. Em atividades de predição, a partir de dados de treinamento previamente rotulados, o propósito é construir um modelo que possa ser capaz de prever uma classe. Sendo assim, os modelos de AM utilizados seguem o aprendizado supervisionado, onde

há o conhecimento da classe de saída desejada e essa informação é utilizada no treinamento. Tarefas de descrição tem como finalidade explorar um conjunto de dados, não fazem uso da classe de saída, seguindo assim o aprendizado não supervisionado.

A Figura 4 representa um exemplo de hierarquia de aprendizado indutivo de acordo com o tipo de tarefa. O topo apresenta o aprendizado indutivo onde são realizadas as generalizações dos dados, o indutor recebe um conjunto de dados e partir desse conjunto, extrai regras e padrões. Após há a divisão entre aprendizado supervisionado e não supervisionado. Em tarefas supervisionadas, há a divisão dos dados em classificação e regressão. As tarefas não supervisionadas são divididas de modo geral em: agrupamento, que consiste em métodos de clusterização; associação, que compõe em encontrar padrões e sumarização entre conjunto e dados (FACELI *et al.*, 2011).

Figura 4 – Hierarquia de aprendizado

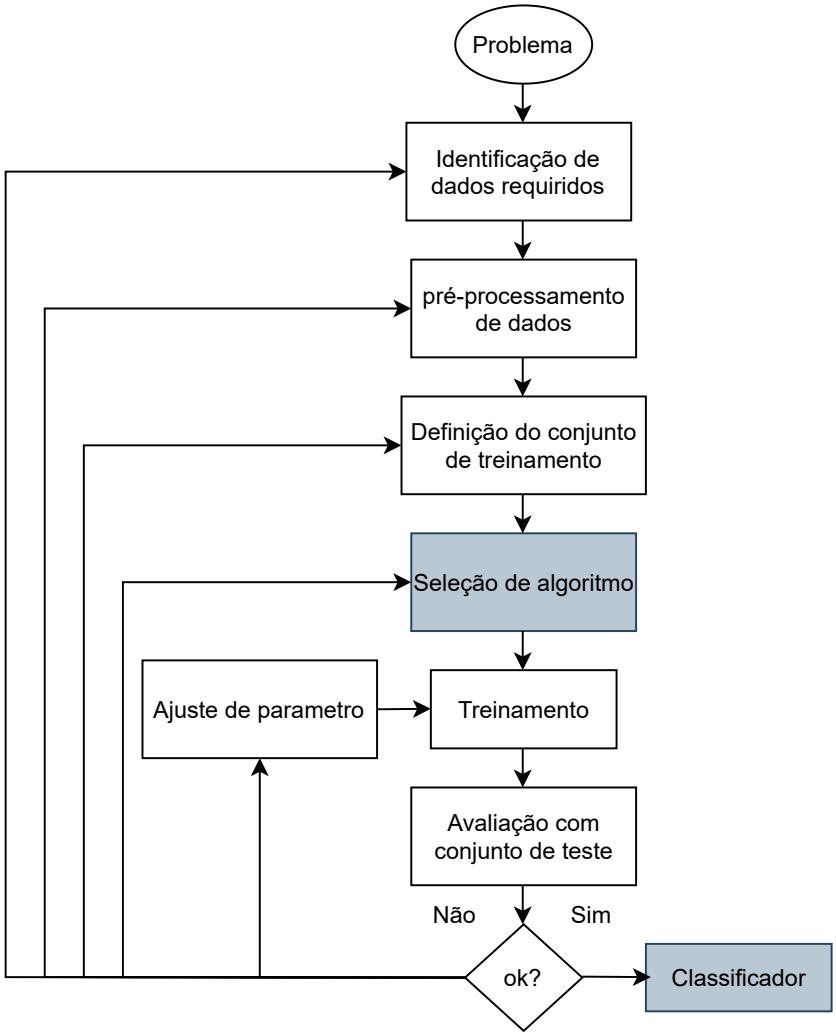


Fonte: Adaptado de Faceli *et al.* (2011).

De acordo com Osisanwo *et al.* (2017), os algoritmos de aprendizado de máquina são organizados em uma taxonomia com base no resultado desejado do algoritmo. A aprendizagem supervisionada consiste em analisar dados previamente rotulados com o fim de gerar um modelo de classificação capaz de produzir rótulos para instâncias não rotuladas. Esse tipo de aprendizado é aplicado em problemas de classificação e regressão porque o objetivo é geralmente fazer com que o computador aprenda um sistema de classificação já criado. O processo de aplicar aprendizado de máquina supervisionado a um problema do mundo real é descrito na Figura 5.

A partir do problema a ser solucionado, há o pré-processamento para a identificação dos dados. Em seguida é realizada a definição do conjunto de treinamento e teste. Após, o conjunto de treinamento é utilizado para a construção de modelos preditivos e o conjunto de teste, para a avaliação dos mesmos. Caso a avaliação esteja satisfatória é realizada a classificação. Caso contrário há a necessidade de reavaliação novamente para verificação de quais passos poderão ser corrigidos.

Figura 5 – Processos de aprendizado supervisionado



Fonte: Adaptado de Faceli *et al.* (2011).

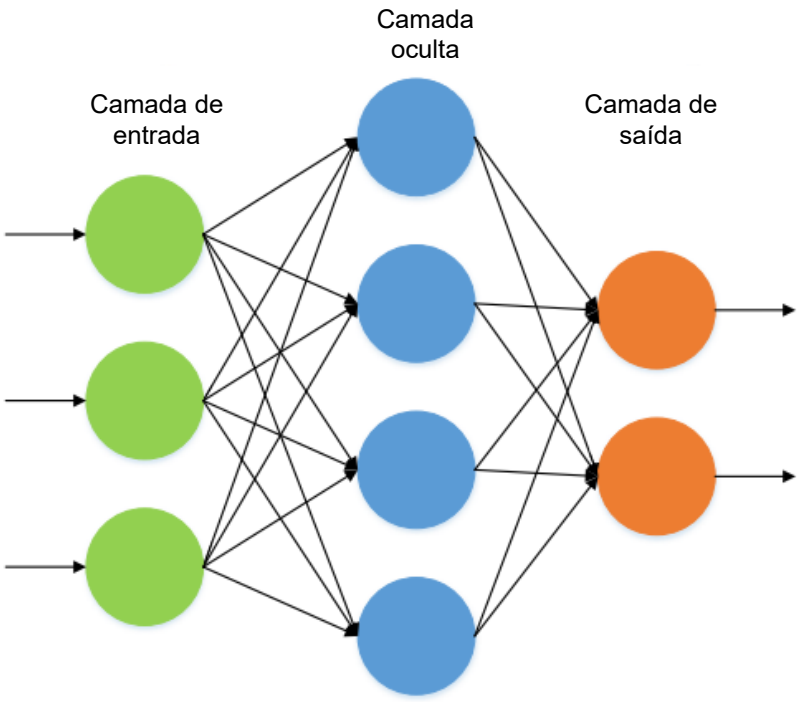
2.3.1 MÉTODOS DE CLASSIFICAÇÃO

Métodos de classificação são estruturados sobre diversos conceitos e teorias. Alguns são fundamentados em árvores de decisão, enquanto outros em redes neurais, teorema de Bayes, funções matemáticas, regras, regressões, entre outros. O desempenho de cada classificador pode variar de acordo com o conjunto de dados utilizado (RODRIGUES; CONCI; LIATSIS, 2018).

As Redes Neurais Artificiais, do inglês *Artificial Neural Networks* (RNAs), são sistemas de processamento computacional inspirados na atuação de sistemas nervosos biológicos, como o cérebro humano. As RNAs são compostas principalmente por um grande número de nós computacionais interconectados, relacionado aos neurônios, dos quais trabalham entrelaçados em uma forma distribuída para aprender coletivamente com a entrada, a fim de otimizar sua saída (O'SHEA; NASH, 2015).

De acordo com O'Shea e Nash (2015), a estrutura básica de uma RNA pode ser modelada conforme mostrado na Figura 6. A entrada é carregada geralmente na forma de um vetor multidimensional, onde será distribuído para as camadas ocultas. As camadas ocultas são então os tomadores de decisões, avaliam como uma mudança estocástica em si mesmo pode melhorar ou prejudicar o resultado final, e isso é referido como o processo de aprendizado.

Figura 6 – Exemplo de rede neural



Fonte: Adaptado de Mingzhe Chen *et al.* (2019).

Nos últimos anos, houveram avanços na construção de classificadores para detecção e reconhecimento de imagens em vários conjuntos de dados utilizando algoritmos de aprendizado de máquina. O aprendizado profundo, do inglês *Deep Learning*, mostrou melhoria na precisão em vários conjuntos de dados. Os algoritmos de Deep Learning são uma implementação de redes neurais artificiais com múltiplas camadas ocultas para imitar as funções do córtex cerebral humano. Esses algoritmos são representações de redes de neurônios profundos (CHAUHAN; GHANSHALA; JOSHI, 2018).

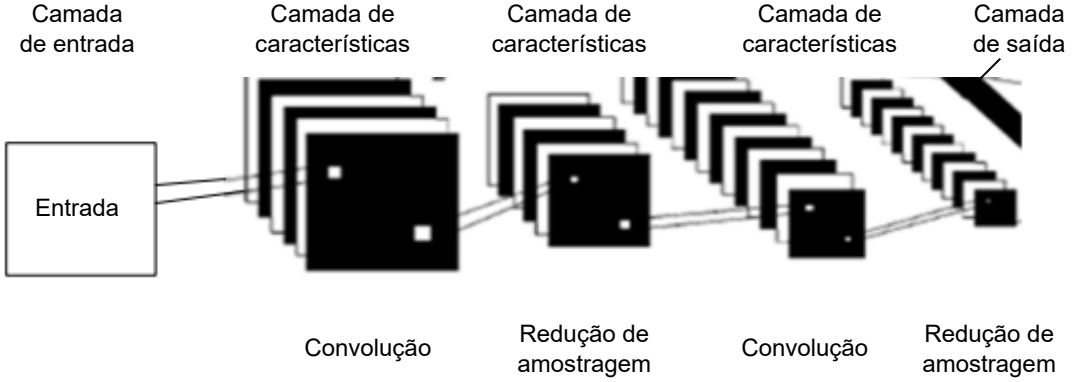
De acordo com Dong, Ping Wang e Abbas (2021), Deep learning nada mais é do que muitos classificadores trabalhando juntos, que são baseados em regressão linear seguida de algumas funções de ativação. Sua base é a mesma da abordagem tradicional de regressão linear estatística A única diferença é que existem muitos neurônios artificiais no aprendizado profundo e apenas um nó é chamado de regressão linear no aprendizado estatístico tradicional. Um nó classificador é conhecido como unidade neural. Outro ponto de contraste precisa ser notado é

que no aprendizado profundo existem muitas camadas entre a entrada e a saída. Uma camada pode conter milhares de neurônios. As camadas que estão entre a entrada e a saída conhecidas como camadas ocultas e os nós são conhecidos como nós ocultos.

Um fator importante em uma rede neural é a função de ativação, inspirada pela descarga neural humana. As funções de ativação são usadas para gerar relações não lineares entre a entrada e a saída. Essa não linearidade, combinada com neurôrios artificiais organizados em camadas, imita o cérebro humano como estrutura, e é por isso que é chamado de rede neural. Existem diversas funções de ativação como por exemplo a relu, e elas são responsáveis por transformar e abstrair os dados em um plano mais classificável. Geralmente, os dados são agrupados muito bem, é o trabalho da função de ativação que transforma os dados em um plano diferente que ajuda a observar os efeitos de diferentes dimensões em dado problema (DONG; WANG, P.; ABBAS, 2021).

Atualmente, redes neurais de convolucionais, do inglês *Convolution Neural Network* (CNN) é um tema alta no campo da análise de dados de voz e reconhecimento de imagem. São algoritmos de aprendizado profundo capazes de lidar com milhões de parâmetros, inserindo imagens 2D e convolucionando-a com filtros/kernel assim produzindo as saídas desejadas (CHAUHAN; GHANSHALA; JOSHI, 2018). Como uma rede neural multicamada, cada camada na estrutura da rede neural de convolução é composta por vários planos bidimensionais e cada plano possui neurônios independentes. As conexões esparsas são utilizadas entre as camadas. Isso significa que o neurônio em cada mapa de características conecta apenas os neurônios em uma pequena área na parte superior, em vez da rede neural tradicional. Um típico modelo da rede neural de convolução é mostrado na Figura 7 (DONG; WANG, P.; ABBAS, 2021).

Figura 7 – Modelo de rede neural convolucional



Fonte: Adaptado de Dong, Ping Wang e Abbas (2021).

De acordo com Dong, Ping Wang e Abbas (2021), estrutura da rede neural de convo-

lucional depende principalmente do peso compartilhado, no campo de experiência local e no subcoletor para garantir a invariância dos dados de entrada. Esses fatores podem ser explicados da seguinte forma, a primeira camada oculta contém seis mapas de características. Cada mapa de características corresponde a uma pequena caixa na camada de entrada. O mapa de características precisa estar acumulado na primeira camada. Cada camada de convolução geralmente é composta por vários mapas de características e o peso das características é o mesmo, que pode reduzir o número de seus próprios parâmetros. Cada camada de convolução tem uma camada de redução de amostragem que realiza a média local para reduzir a sensibilidade associada com a deformação e a translação do resultado.

Como a estrutura da rede neural de convolução é principalmente alternadamente composto de camada convolução e camada de Redução de amostragem, com a redução da resolução espacial, o número de mapas de características também está aumentando. A primeira fase é de treinamento, que consiste em três etapas. Selecionar as amostras de acordo com o conjunto de amostras fornecido aleatoriamente. Colocar as amostras como dados iniciais na rede. Calcular os dados de saída correspondentes. A segunda fase, é a fase de propagação, consiste em duas etapas. Calcular a diferença entre as informações de dados ideais e as informações de dados de saída. Ajustar a matriz de pesos de acordo com a minimização do método de erro para a transmissão reversa (DONG; WANG, P.; ABBAS, 2021).

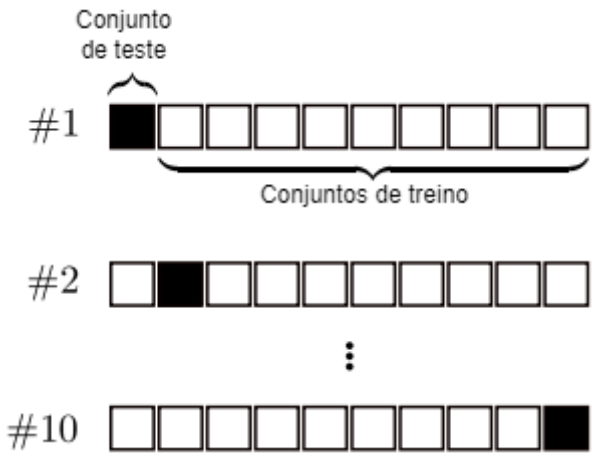
2.3.2 AVALIAÇÃO DE MODELOS DE CLASSIFICAÇÃO

A validação cruzada, do inglês *cross-validation*, é um método de re-amostragem de dados amplamente utilizado para avaliar a capacidade de generalização de modelos preditivos. O conjunto de dados disponível para construir e avaliar um modelo preditivo é denominado como o conjunto de aprendizagem. Métodos de subamostragem aleatória são utilizados para gerar o conjunto de treinamento, e o conjunto de teste, a partir do conjunto de aprendizagem.

O k-fold é um método de validação cruzada, onde o conjunto de treinamento é particionado em k subconjuntos disjuntos de tamanho aproximadamente igual. O modelo é treinado utilizando $k - 1$ subconjuntos, que, juntos, representam o conjunto de treinamento. Então, o modelo é aplicado ao subconjunto restante, denominado como o conjunto de teste e o desempenho é medido. Este procedimento é repetido até que cada um dos k subconjuntos tenha sido como conjunto de teste. O desempenho do modelo é resultante de medidas estatísticas (por exemplo, média e desvio-padrão) calculadas a partir dos k valores de acurácia ou erro gerados na

validação cruzada. (BERRAR, 2019). A figura 8 mostra um exemplo de validação cruzada com $k = 10$, ou seja, validação cruzada de dez *folds*.

Figura 8 – Validação cruzada com $k = 10$



Fonte: Adaptado de Berrar (2019).

No conjunto #1, o primeiro subconjunto serve como conjunto de teste e os nove subconjuntos restantes servem como conjunto de treinamento. No conjunto #2 o segundo subconjunto é o conjunto de teste e os subconjuntos restantes são o conjunto de treinamento e assim por diante. O conjunto de dados é dividido aleatoriamente em dez subconjuntos separados, cada um contendo aproximadamente 10% dos dados. O modelo é treinado utilizando o conjunto de treinamento e, em seguida, avaliado por meio do uso do conjunto de teste (BERRAR, 2019).

2.3.3 TRABALHOS RELACIONADOS

Atualmente, de acordo com a revisão bibliográfica realizada neste trabalho, não existem trabalhos sobre a edição de vídeos visando a retirada de momentos sem falas utilizando aprendizado de máquina, sendo esse o problema do presente trabalho.

Um exemplo de ferramenta ara facilitar a edição de vídeos é o Silver (CASARES *et al.*, 2002), que fornece seleções inteligentes de clipes de vídeo, bem como visualizações abstratas de edição de vídeo, usando metadados dos vídeos. O Silver oferece muitas visualizações da composição do vídeo. Essas visualizações permitem que o usuário navegue no vídeo exibindo-o em espaço ao invés de tempo, para visualizar e manipular o áudio e vídeo separadamente. Uma ferramenta mais recente é o Roughcut (LEAKE *et al.*, 2017), que permite a edição computacional para cenas orientadas por diálogo usando a entrada do usuário de diálogo para a cena, gravações brutas e edição de expressões idiomáticas. O sistema quebra automaticamente um roteiro de

entrada em linhas de diálogo faladas por cada personagem.

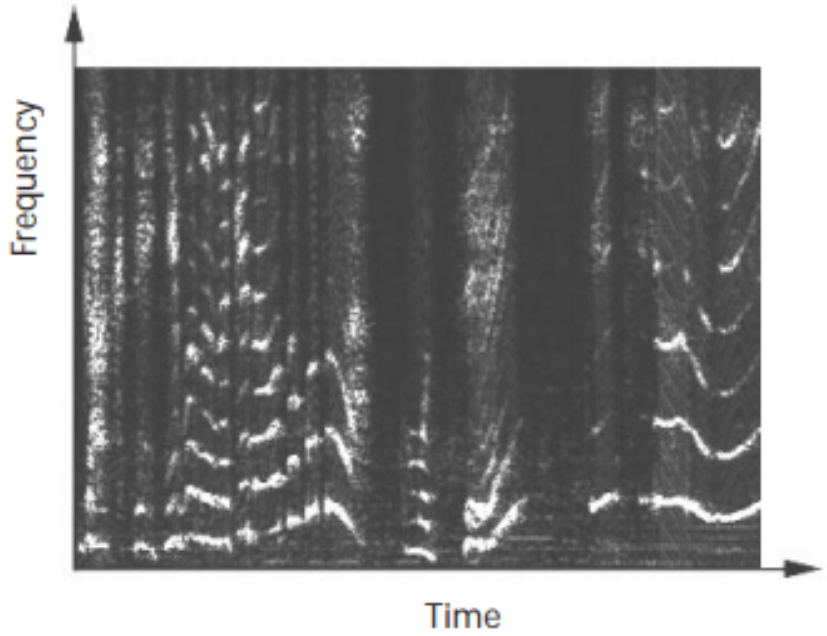
A queda nos custos de hardware provocou uma explosão na captura de vídeos por usuários domésticos. Vídeos caseiros, muitas vezes contêm pontos oscilantes à medida que o operador da câmera muda o assunto, ou até mesmo ficam muito longos por conta de cenas que podem ser descartadas. Isso pode levar a vídeos que não são tão interessantes de assistir. (WANG, T. *et al.*, 2009) propuseram um algoritmo para dar vida à vídeos amadores, editando imagens de vídeo brutas. Levando em consideração as operações de corte para o vídeo ser encurtado, zoom para dar foco e visualização que se move para dar foco no assunto. Essas operações podem ser aplicadas ao vídeo de origem, com parâmetros apropriados e em uma sequência específica, para produzir um vídeo editado utilizando um *framework* de programação genética. É um método de otimização evolutiva, semelhante ao algoritmo Genético. Utiliza soluções de árvores de análise em um espaço de busca de dimensão fixa, assim se tornando adequado ao problema da edição de vídeo, uma vez que o número e ordem das operações de edição pode variar entre as sequências de vídeo. Além disso, algoritmos evolutivos são adequados para grandes espaços de busca nos quais a combinação de soluções distintas, mas localmente ótimas.

Embora imagens sejam tradicionalmente usadas para edição de vídeos, os áudios são uma rica fonte de informação. Grande parte dos estudos sobre fala visa melhorar os sistemas de reconhecimento de fala. Kimber, Wilcox *et al.* (1997) propuseram um modelo oculto de Markov do inglês *Hidden Markov Model* (HMM), algoritmo baseado para classificar música e fala, bem como outras classes de som. A classe é treinada utilizando a estimativa de máxima probabilidade, a segmentação é obtida utilizando o algoritmo de Viterbi para determinar a estimativa de máxima probabilidade através desta rede, dada uma sequência de vetores de características, e observando aqueles momentos em que o estado passa entre estados associados a diferentes classes. Os modelos são treinados inicialmente com alguns dados rotulados para cada classe de áudio. Isso pode ser obtido rotulando manualmente uma parte da gravação de áudio. Os modelos são então treinados e utilizados para segmentar o sinal de áudio.

A dificuldade em lidar com fontes de áudio mistas tem, até agora, dificultado o uso de informações de áudio no manuseio de vídeos. Portanto, poucos têm tentado lidar com o tipo de vídeos que recebemos transversalmente em situações cotidianas. Minami *et al.* (1998) propuseram algoritmos de classificação de áudio capazes de detectar música e fala de forma independente, mesmo com ruídos de fundo. Podemos organizar os segmentos de áudio de um vídeo em da mesma forma que organizamos imagens. A ideia para detecção de música vem do método de Hawley, que considera energia estável onde o espectro atinge o pico de uma

característica. O algoritmo de detecção de fala primeiro calcula a potência espectro usando a transformada rápida de Fourier. A Figura 9 mostra um exemplo de espectrograma de som da fala. A característica mais aparente, o espaçamento igual listras, resultado da harmonia do som vozeado estrutura. Uma série de listras na direção do tempo parecem combinar claramente com a emissão de vogais, então detectar essas listras nos permite identificar se o segmento de som inclui fala.

Figura 9 – Espectrograma de som de fala



Fonte: Minami *et al.* (1998).

O método trata o espectrograma de som como uma imagem em escala de cinza e calcula a localização dos picos aplicando um operador de detecção de borda. A resposta de um espectro de linha é dada pelo valor máximo em um determinado espaço e posição. A potência do espectro quando subtraído a meio caminho entre, evita a resposta excessiva do pente ao ruído de banda larga. Se a resposta exceder o limiar, o segmento de som é reconhecido como som vozeado (MINAMI *et al.*, 1998).

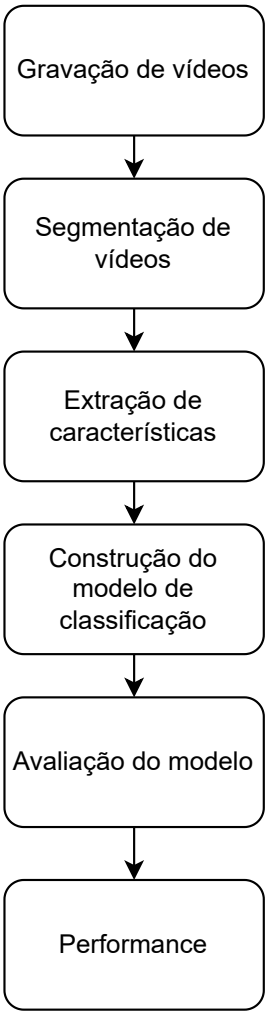
Tun *et al.* (2020) propuseram a extração de características de som utilizando coeficientes cepstrais de frequência de Mel. Os autores extraíram características de sinais de som de entrada para a identificação de fala. Os procedimentos para a extração foram: o enquadramento do sinal para a separação em *frames*, aplicação do janelamento de *hamming*, transformada rápida de Fourier, cálculo de log de energia do MFCC e a transformação discreta do cosseno, assim por fim obtendo os coeficientes de cepstrais de frequência de Mel. Esta abordagem também foi utilizada

para a extração de características de som nesse trabalho em questão, onde estas características servem de entrada para o desenvolvimento de um modelo de classificação.

3 METODOLOGIA

Este capítulo apresenta as etapas de desenvolvimento do trabalho bem como extração dos trechos do vídeo, extração de características, construção do modelo de classificação, métodos para a avaliação do modelo e testes. A Figura 10 apresenta o fluxograma de etapas realizadas para a resolução do problema de segmentação de vídeos por trechos de fala.

Figura 10 – Fluxograma de etapas



Fonte: Autoria própria (2022).

A entrada utilizada se dá por diversos vídeos de autoria própria, ao qual são cortados de forma aleatória a fim de se obter diversos trechos. O áudio de cada trechos de vídeo é extraído assim se obtendo a base de dados para posteriormente realizar a etapa de extração de característica. Com as características extraídas de cada trecho de vídeo, é realizada a etapa de treinamento para a construção do modelo de classificação. Por fim, a verificação de performance do modelo é realizada. Cada etapa é descrita detalhadamente nas próximas seções.

3.1 GRAVAÇÃO DE VÍDEOS

Nessa etapa, foi realizada a gravação de diversos vídeos, todos de autoria própria, a partir de um dispositivo celular. Estes vídeos são no formato de *vlog*, onde a imagem de gravação é do rosto da pessoa. Ao longo do vídeo, há diversos momentos de fala como momentos sem fala. Também, é importante destacar que a minutagem dos vídeos é variada.

3.2 TRECHOS DE VÍDEO

Em cada vídeo são realizados diversos cortes de forma aleatória, afim de se obter vários trechos, sendo cada um com minutagem variada e é previamente classificado. A nomenclatura de cada trecho se dá pela seguinte forma (trecho)-(video_id)-(start)-(end)-(class_id), onde:

- Trecho: nome do trecho de cada vídeo levando em conta o vídeo original;
- Video_id: id do vídeo original;
- Start: minutagem inicial do trecho levando em conta a minutagem do vídeo original;
- End: minutagem final do trecho levando em conta a minutagem do vídeo original;
- Class_id: id de classificação do trecho.

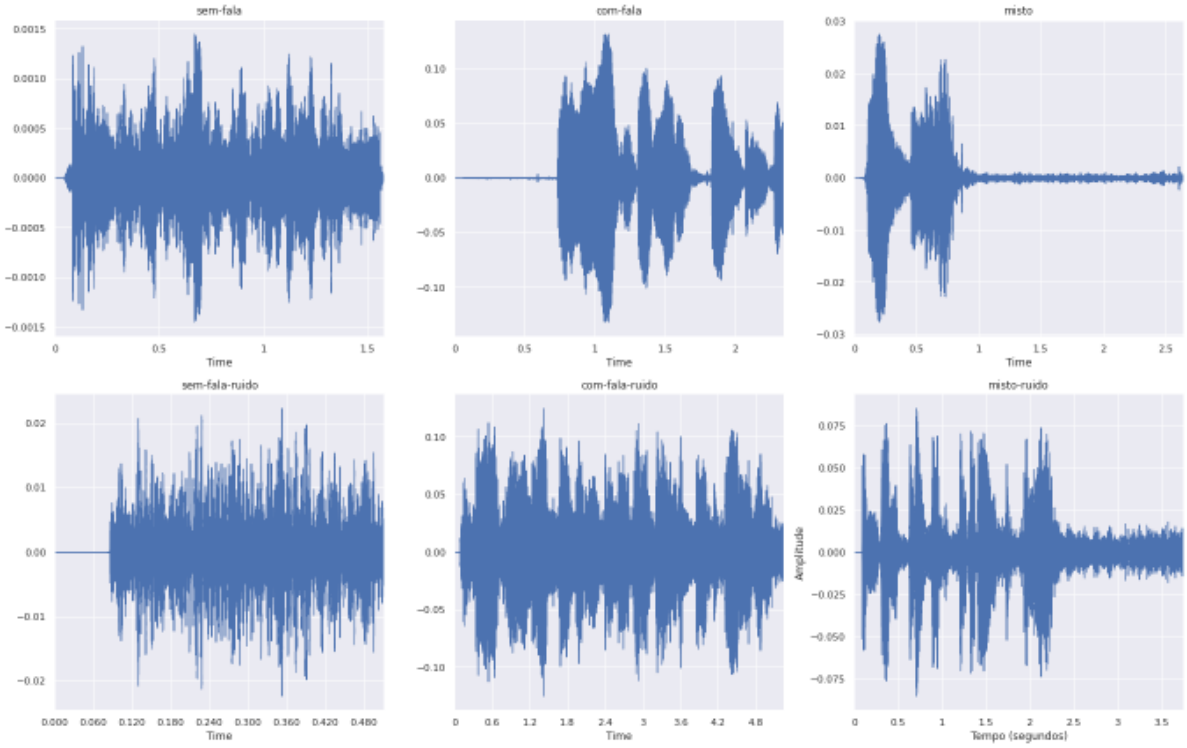
A pesquisa foi desenvolvida através de dados supervisionados, ou seja, a base de dados já está previamente classificada. A classificação foi realizada de forma manual analisando cada trecho de vídeo que foram classificados como: 0 para trechos que não contém fala alguma; 1 para trechos de vídeo que contém apenas momentos de fala; 2, como trechos mistos onde pode conter tanto momentos sem fala quanto momentos com fala; 3, para trechos sem fala que contém ruídos de fundo; 4 para trechos com fala que contém ruídos; e 5, como trechos mistos com ruídos de fundo onde pode conter tanto momentos sem fala quanto momentos com fala.

- 0: sem-fala
- 1: com-fala
- 2: misto
- 3: sem-fala-ruído
- 4: com-fala-ruído
- 5: misto-com-ruído

3.3 EXTRAÇÃO DE ÁUDIO

A partir de cada clipe de vídeo, houve a extração do áudio. Essa extração foi realizada com a utilização da biblioteca Python ffmpy [1], que contém diversos recursos para a extração do arquivo de áudio. Neste projeto foi utilizada a saída o formato wav (*waveform audio file format*), que é um formato-padrão de arquivo de áudio. A Figura 11 mostra um exemplo de gráfico de onda para cada classe.

Figura 11 – Exemplo gráfico de onda para cada classe



Fonte: Autoria própria (2022).

Os gráficos de onda são utilizados para representar o domínio do tempo de um sinal, mostrando a amplitude da onda sonora mudando conforme o tempo. Em cada trecho de vídeo pode ser realizada a extração de características.

3.4 EXTRAÇÃO DE CARACTERÍSTICA

Diversas características podem ser computadas para representação de sinais para um formato de entrada adequado e a aplicação de algoritmos de aprendizado de máquina. No geral,

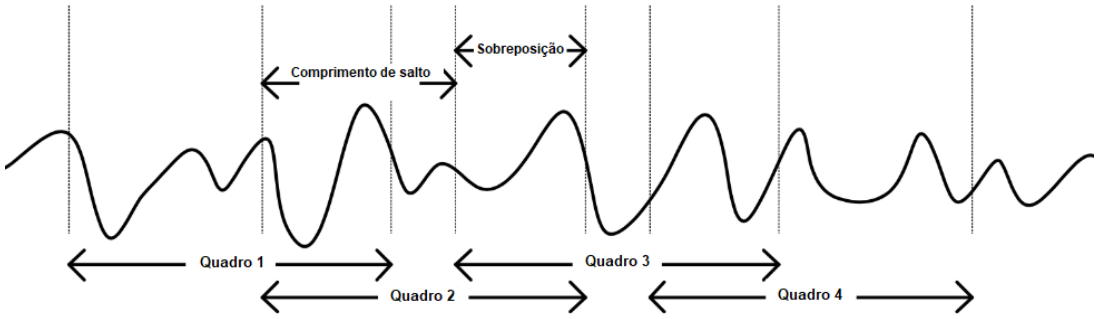
[1] Detalhes sobre a biblioteca podem ser acessados em: ffmpy.

essas características são extraídas no domínio de tempo, domínio de frequência, domínio de tempo-frequência e por abordagem não linear (OLIVA; ROSA, 2021).

Em cada trecho de áudio é aplicado a transformada de Fourier de curta duração, do inglês *Short Time Fourier Transform* (STFT), uma técnica matemática utilizada para transformar o sinal de tempo discreto do domínio do tempo em sua frequência. A STFT analisa apenas uma pequena seção do sinal de cada vez, utilizando a técnica de janelamento de sinal (HIBARE; VIBHUTE, 2014).

O janelamento é utilizado para encontrar o comprimento ideal da janela para o processo de extração de características. Os trechos de áudio são divididos em várias janelas de comprimento de quadro constante e processados cumulativamente. Esse comprimento é chamado de 'comprimento da janela', do inglês *window length*. Para preservar a continuidade durante o processamento faz-se necessário deixar alguns quadros de janelas subsequentes se sobreporem e a contagem real única de quadros processados por janela é conhecida como 'comprimento do salto' do inglês *hop length* (VELAYUDHAM, 2020). Um processo de janelamento é demonstrado na Figura 12.

Figura 12 – Processo de janelamento



Fonte: Adaptado de Velayudham (2020).

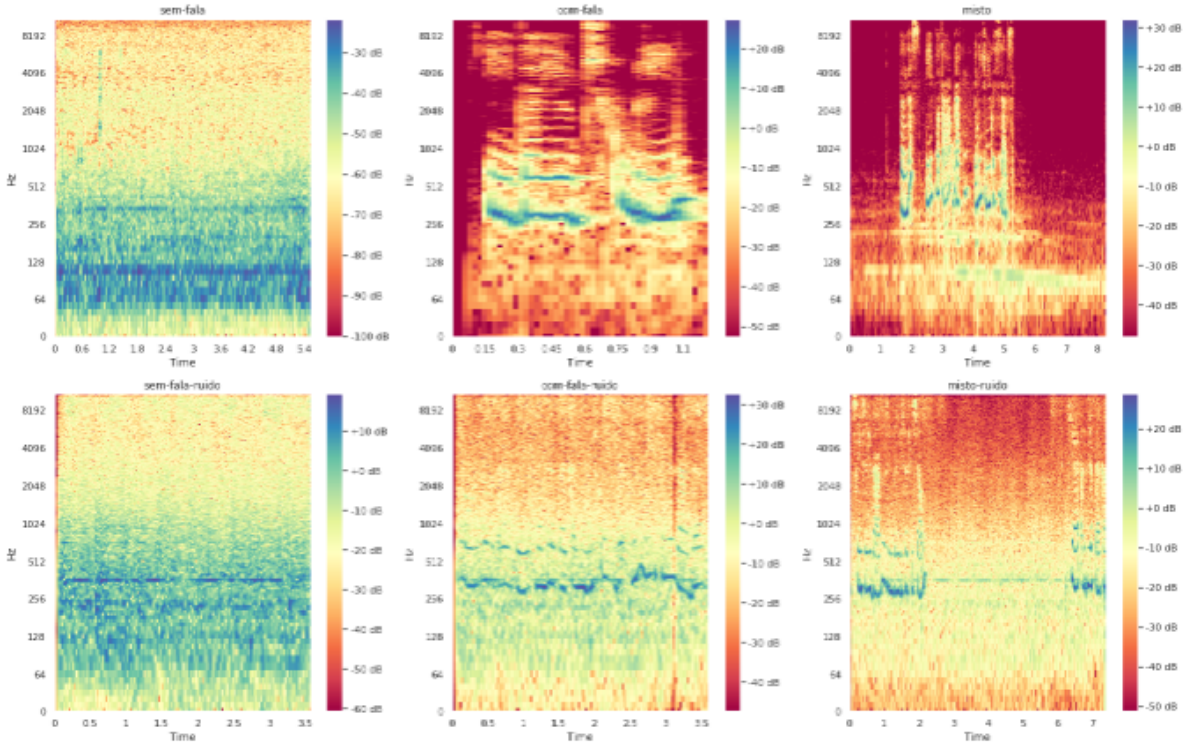
A janela de um determinado comprimento (número de FFT) desliza sobre o sinal de áudio 1d, ou seja, ele analisa um segmento (ou quadro) mais curto de cada vez, calcula características para este segmento e passa para o próximo segmento. Esses segmentos geralmente se sobrepõem. A distância entre dois desses segmentos é denominada comprimento de salto e é especificada em número de amostras. Pode ser idêntico a número de FFT, mas muitas vezes o comprimento de salto é metade ou até mesmo um quarto de número de FFT. Ele permite o controle da resolução temporal de seus recursos.

A transformada rápida de Fourier transforma o sinal de tempo discreto do domínio do tempo em sua frequência (HIBARE; VIBHUTE, 2014) utilizando a Equação (1).

$$\overline{X}_n = \sum_{n=0}^{N-1} (e^{-j2\pi kn/N}) \cdot \overline{X}_n \tag{1}$$

Esses sinais representam uma decomposição de sinal em relação aos componentes senoidais (HAQUE, 2012). O resultado dessa etapa são números complexos que consistem em parte real e imaginária. Foi utilizado para caracterizar o espaço do padrão de extração de características de som (TUN *et al.*, 2020). A Figura 13 representa um exemplo de espectrograma STFT para cada classe.

Figura 13 – Exemplo STFT para cada classe



Fonte: Autoria própria (2022).

Em resumo a STFT para geração de um espectrograma trabalha da seguinte maneira: separa o sinal em quadros/janelas; para cada quadro calcula a transformada de Fourier; em seguida; cria a matriz cujas colunas são as transformações; e por fim, traça o mapa de calor desta matriz.

Após a aplicação da STFT, podemos aplicar diversos métodos para extração de características do som. Nesse presente trabalho será aplicado o método de Coeficiente Cepstral de Frequência de Mel por ser uma das técnicas que apresenta melhores resultados para o estudo de características de som.

Para a extração de características utilizando o MFCC, é necessário o cálculo dos coeficientes que representam a frequência cepstral. Esses coeficientes são utilizados com base na transformada de cosseno linear da potência logarítmica do espectro na escala não linear de frequência de Mel (HIBARE; VIBHUTE, 2014). A Equação (2) mostra a conversão da frequência

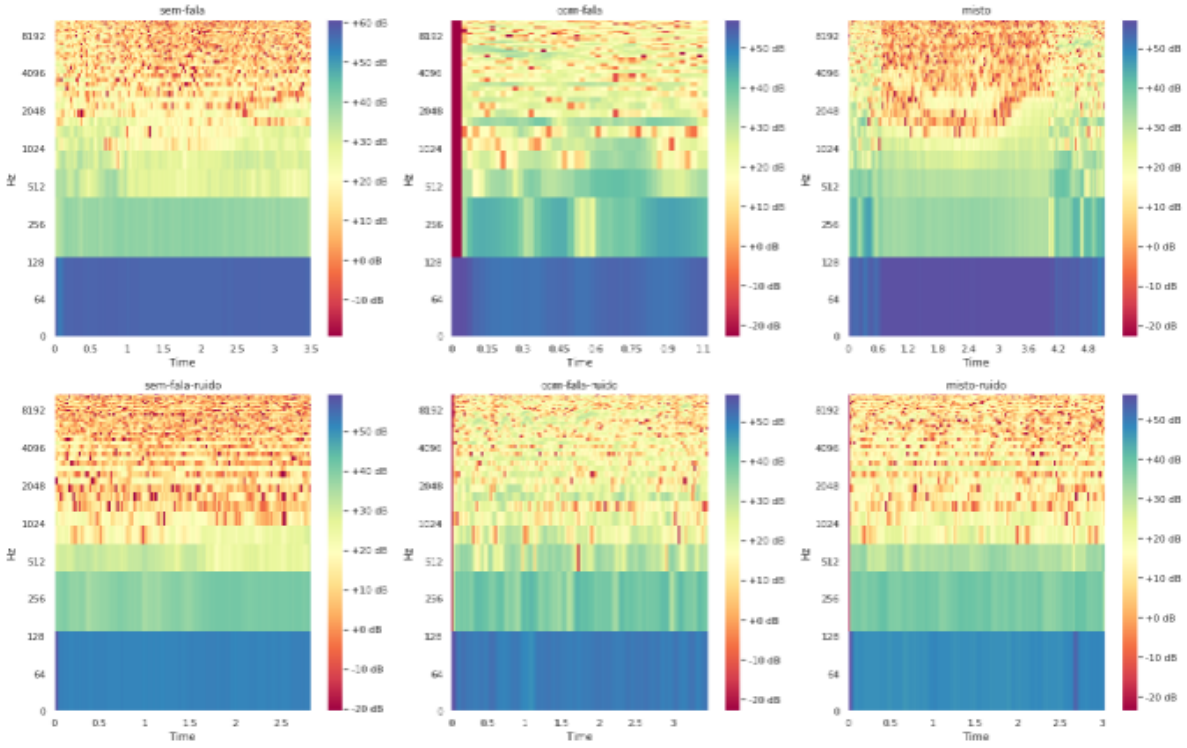
normal f para a escala de Mel m.

$$m = 2595log_{10}(1 + \frac{f}{100}) \quad (2)$$

O Mel é uma unidade de altura e a escala de frequência de Mel é a aproximação da frequência linear (1KHz) então próximo ao logarítmico para frequências mais altas. O cálculo da energia do MFCC é realizado pelo logaritmo da magnitude quadrada de saída do banco de filtros Mel. Essa operação comprime a faixa dinâmica de valores (TUN *et al.*, 2020).

Um exemplo de espectrograma MFCC para cada classe é demonstrado na Figura 14.

Figura 14 – Exemplo MFCC para cada classe



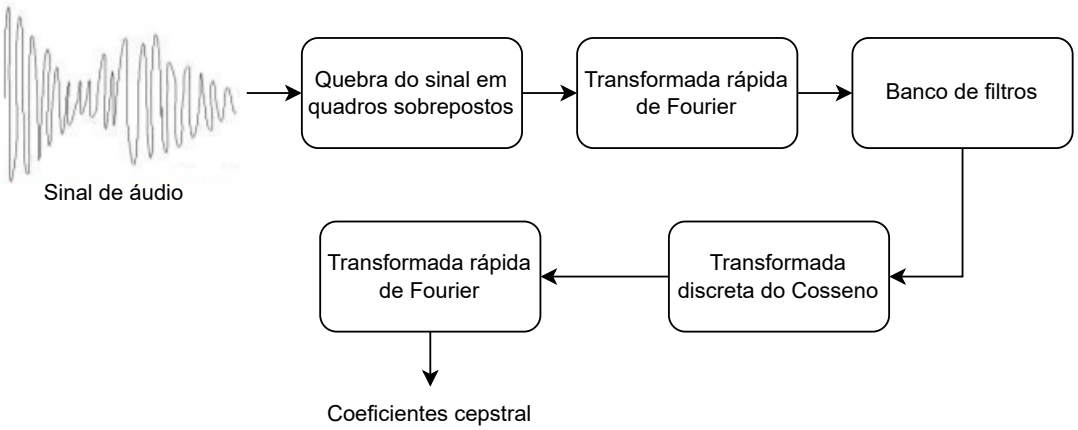
Fonte: Autoria própria (2022).

A Figura 15 representa o fluxo e etapas para o calculo do MFCC.

A entrada é o sinal de áudio, ao qual é aplicado o janelamento e, em seguida ,a transformada rápida de Fourier gerando o banco de Filtros que refere-se aos filtros mel, e seguida é aplicado o calculo da transformada discreta do cosseno e novamente a transformada rápida de Fourier gerando assim os coeficientes cepstral que são as características MFCCs.

Por fim, é aplicado normalização das características MFCCs extraídos de cada arquivo de áudio utilizando a técnica de Normalização Média Cepstral, do inglês *Cepstral Mean Normalization* (CMN). Isto reduzirá o ruído e variações que ocorrem nos canais de cada arquivo

Figura 15 – Etapas calculo do MFCC



Fonte: Adaptado de Nair (2018).

de áudio, pois as condições do canal são diferentes para cada arquivo. Assim, cada característica de áudio extraído é padronizado com sua própria média.

3.5 CONSTRUÇÃO DE MODELOS DE CLASSIFICAÇÃO

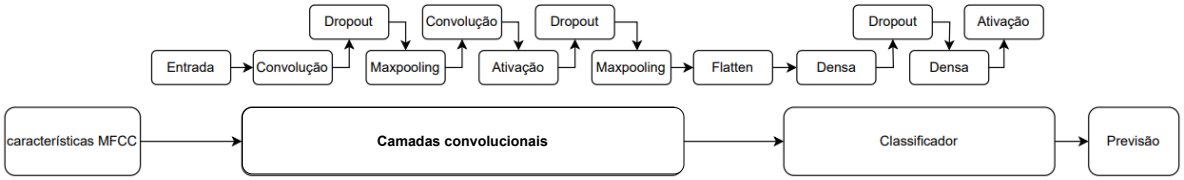
Os espectrogramas extraídos dos trechos de áudio são como imagens 2D, desta forma podemos utilizar técnicas de classificação de imagens. A técnica de classificação escolhida é a rede neural convolucional pois é um dos melhores métodos de construção de modelos de classificação. A rede neural convolucional contêm os seguinte parâmetros:

- Ativação: aplicação de funções de ativação;
- MaxPooling: camada responsável pela extração das principais características;
- Convolução: uma rede neural convolucional que realiza a convolução ao longo de apenas uma dimensão;
- Flatten: camada para transformar a matriz em vetor;
- Densa: quando um neurônio de uma camada está ligado a todas os outros neurônios das outras camadas;
- Dropout: técnica de regularização para diminuição do overfitting;

A arquitetura da rede neural convolucional é mostrada na Figura 16, foi definida a partir de testes realizados para se obter o resultado desejado.

A rede neural recebe como entrada as características extraídas. A primeira camada convolucional será de uma dimensão, a função de ativação utilizada é a relu, função padrão aplicada em camadas ocultas. Logo após conterá uma camada de *dropout* para evitar a ocorrência

Figura 16 – Arquitetura rede neural convolucional



Fonte: Autoria própria (2022).

de *overfitting*. Na sequência, há a camada de *maxpooling* para obtenção dos valores mais significativos das características. Há uma nova camada convolucional com a função de ativação relu e outra camada de *dropout*. Há uma camada de *flatten* para a conversão do resultado que está em formato de matriz para o formato de vetor.

Na parte de classificação há uma camada densa juntamente com mais uma camada de *dropout*. Por fim, há mais uma camada densa de saída seguida por uma função de ativação *softmax* ao qual gera uma probabilidade para cada uma das classes.

3.6 AVALIAÇÃO DE MODELO

O conjunto de dados foi dividido em conjunto de treinamento onde é realizada a tarefa de aprendizagem e conjunto de teste para avaliação das características dos modelos.

Para a verificação da performance dos modelos, foi utilizado a validação cruzada, que é um método de re-amostragem de dados para avaliar a capacidade de generalização de modelos preditivos. O método de validação cruzada que será utilizado é o k-fold, onde o conjunto de treinamento é particionado em k subconjuntos disjuntos de tamanho aproximadamente igual.

Os códigos e a base de dados estão disponíveis no GitHub e podem ser acessado em "Segmentação de vídeos por trechos de fala".

4 CLASSIFICAÇÃO DE TRECHOS DE ÁUDIO

Este capítulo apresenta detalhes das etapas realizadas para a classificação de trechos de vídeo. Esta classificação permitirá que trechos sem falas sejam descartados da edição final de um vídeo.

4.1 IMPLEMENTAÇÃO

Para a montagem da base de dados utilizada para treinamento do modelo faz-se necessário o corte de cada vídeo em diversos trechos este corte foi realizado de forma aleatória utilizando um editor de vídeos. Em cada trecho dos vídeos foi realizada a extração do áudio em arquivos no formato .wav utilizando a biblioteca python ffmpy, como mostrado no Código-fonte 1.

Código-fonte 1 – Extração do arquivo de áudio

```
1  !pip install ffmpy
2
3  import os
4  import ffmpy
5
6  inputdir = '/videos'
7
8  for filename in os.listdir(inputdir):
9      actual_filename = filename[:-4]
10     if(filename.endswith(".wmv")):
11         os.system('ffmpeg -i {} -acodec pcm_s16le -ar 16000 {}.wav'.format(
       filename, actual_filename))
12     else:
13         continue
```

Fonte: Autoria própria (2022).

Os trechos de vídeo estão inicialmente no diretório da pasta vídeos e tem formato inicial .wmv. Cada trechos é transformado em um arquivo de áudio do tipo .wav.

Ao todo, o conjunto de dados contém 587 trechos de áudio dos quais foram classificados previamente dentro das 6 classes definidas: trechos sem fala, com fala, misto, sem fala e ruídos, com fala e ruídos ou misto e ruídos. O nome de cada arquivo contém informações significativas, como o identificador do vídeo original e sua classe pertencente. Desta forma faz-se necessário a extração dos metadados de sua nomenclatura conforme o Código-fonte 2.

Desta forma são extraídos do nome de cada trecho listas que contém os metadados. O nome_list é uma lista que contém informações sobre o nome do trecho de cada vídeo levando em conta o vídeo original. A lista video_list contém o identificador do vídeo original. Start_list é

Código-fonte 2 – Extração metadados

```
nome_list = []
video_id = []
start_list = []
end_list = []
class_id = []
full_path = []

caminho = '/audio_wav/'

for root, dirs, files in os.walk(caminho):
  for file in files:
    try:
      nome = int(file.split('-')[0])
      video = int(file.split('-')[1])
      start = int(file.split('-')[2])
      end = int(file.split('-')[3])
      classe = file.split('-')[4]
      classe= int(classe.split('.')[0])

      nome_list.append(nome)
      video_id.append(video)
      start_list.append(start)
      end_list.append(end)
      class_id.append(classe)

      full_path.append((root, file))
    except ValueError:
      continue
```

Fonte: Autoria própria (2022).

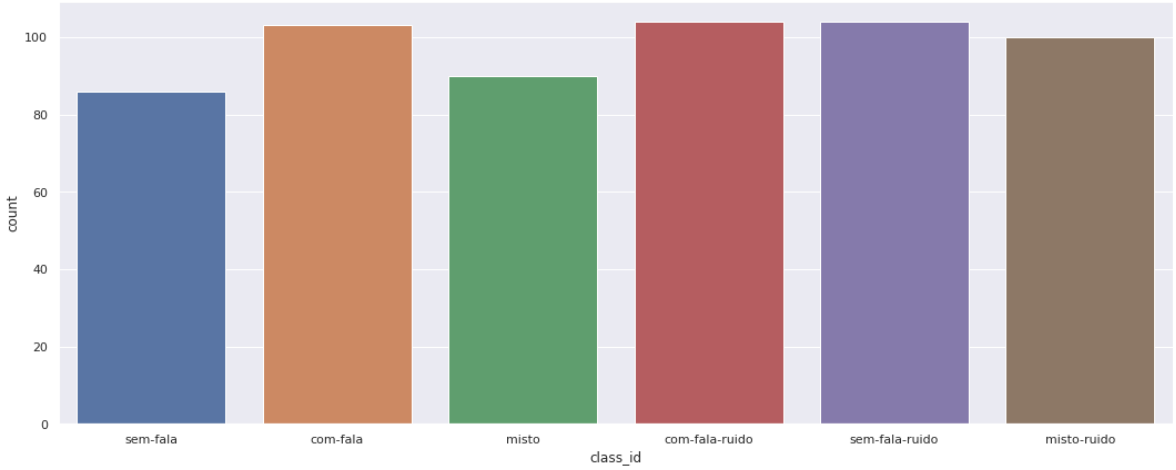
a lista com a informação da minutagem inicial do corte do trechos levando em conta o vídeo original completo e end_list contém a minutagem final do trecho. A lista class_id contém a informação da classe atribuída a cada trecho. Full_path contém o caminho diretório onde se encontra cada trecho.

Com os metadados podemos observar a quantidade de trechos existentes para cada classe, conforme mostrado na Figura 17. Dentre os 587 trechos, tem-se 90 de classe misto, 103 trechos com-fala, 86 sem-fala, 100 trechos misto com ruídos e 104 trechos tanto com fala e ruído como sem fala e ruído.

Foi realizada a extração de características MFCC's de cada trechos áudio do conjunto de dados. Desta forma será possível enviar estas características para treinamento na rede neural. A extração de características foi realizada conforme o Código-fonte 3.

A função feature_extratactor tem como entrada o arquivo de áudio com taxa de amostragem padrão de cada arquivo e a utilização do *kaiser best* para obtenção de melhores resultados. A quantidade de características extraídas é 40 para se obter um melhor resultado no

Figura 17 – Quantidade de trechos para cada classe



Fonte: Autoria própria (2022).

Código-fonte 3 – Extração de características

```
1  def features_extractor(file_name):
2    data, sample_rate = librosa.load(file_name, sr = None, res_type = '
       kaiser_best')
3    mfccs_features = librosa.feature.mfcc(y = data, sr = sample_rate, n_mfcc
       =40)
4    mfcss_features_scaled = np.mean(mfccs_features.T, axis = 0)
5    return mfcss_features_scaled
6
7  extracted_features = []
8  for path in tqdm(df['path'].values):
9    data = features_extractor(path)
10   extracted_features.append([data])
```

Fonte: Autoria própria (2022).

processo de treinamento. Na sequência é aplicado a normalização para reduzir ruídos e variações. Todas as características extraídas estão sendo armazenadas na variável de lista chamada de extracted_features.

Faz-se necessário a divisão entre classe e os atributos previsão para servir de entrada para a rede neural. Desta forma, tem-se a variável X que contém as características extraídas e a variável y ao qual contém as classes de forma categórica, desta forma é necessário sua conversão para valores numéricos. Esta conversão ocorre no Código-fonte 4 na linha 5.

Código-fonte 4 – Divisão entre classe e atributos

```
1  X = np.array(extracted_features_df['feature'].tolist())
2  y = np.array(df['class_id'].tolist())
3
4  labelencoder = LabelEncoder()
5  y = to_categorical(labelencoder.fit_transform(y))
```

Fonte: Autoria própria (2022).

Por fim, ocorre a separação do conjunto de dados em validação, treinamento e testes conforme mostrado no Código-fonte 5.

Código-fonte 5 – Divisão do conjunto de dados

```
1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.5,
      random_state = 1)
2 X_test, X_val, y_test, y_val = train_test_split(X_test, y_test, test_size =
      0.5, random_state=1)
3
4 X_train = X_train[:,:,np.newaxis]
5 X_test = X_test[:,:,np.newaxis]
6 X_val = X_val[:,:,np.newaxis]
```

Fonte: Autoria própria (2022).

Foi utilizada, para realizar esta separação do conjunto, a função train_test_split do *sklearn*. A criação da estrutura da rede neural convolucional é mostrada no Código-fonte 6.

Código-fonte 6 – Criação do modelo

```
1  model = Sequential()
2
3  model.add(Conv1D(64, kernel_size=(10), activation='relu', input_shape=(
       X_train.shape[1], 1)))
4  model.add(Dropout(0.4))
5  model.add(MaxPooling1D(pool_size=(4)))
6
7  model.add(Conv1D(128, 10, padding='same',))
8  model.add(Activation('relu'))
9  model.add(Dropout(0.4))
10 model.add(MaxPooling1D(pool_size=(4)))
11
12 model.add(Flatten())
13
14 model.add(Dense(units = 64))
15 model.add(Dropout(0.4))
16 model.add(Dense(units = 6))
17 model.add(Activation('softmax'))
18
19 model.compile(loss='binary_crossentropy', metrics = ['accuracy'], optimizer
       = 'adam')
20 model.summary()
```

Fonte: Autoria própria (2022).

A primeira camada convulacional será de uma dimensão com 64 filtros e o *kernal* responsavél pelos cálculos será igual 10. A função de ativação utilizada é a relu, função padrão utilizada em camadas ocultas. É adicionada uma camada de *dropout* em 40% para evitar a ocorrência de *overfitting*. A camada de *maxpooling* receberá uma matriz 4x4 para realizar os calculos dos maiores valores. A próxima camada convulacional tem 128 filtros com *kernal* 10, função de ativação relu e outra camada de *dropout*. Em sequência, há uma camada de *flatten* para

a conversão de matriz em vetor. Na parte de classificação há uma camada densa com 64 neurônios seguida de uma nova camada de *dropout*, uma camada densa de saída com 10 neurônios e por fim uma camada de ativação com a função *softmax*.

Após a definição da estrutura é realizado o treinamento da rede neural, que foi realizado conforme o Código-fonte 7.

Código-fonte 7 – Treinamento do modelo

```
1 num_epochs = 150
2 num_batch_size = 32
3
4 checkpointer = ModelCheckpoint(filepath = 'saved_models/classification.hdf5
      ', verbose = 1, save_best_only = True)
5
6 start = datetime.now()
7 history = model.fit(X_train, y_train, batch_size = num_batch_size, epochs =
      num_epochs, validation_data = (X_val, y_val), callbacks = [checkpointer
      ], verbose = 1)
8 duration = datetime.now() - start
```

Fonte: Autoria própria (2022).

É utilizado para o treinamento 150 épocas, quanto mais épocas a tendência é que obtenhamos melhores resultados. É enviado para treinamento na rede neural de 32 em 32 trechos de áudio. Será salvo apenas o modelo com melhor resultado para realizar, posteriormente, a etapa de testes.

4.2 RESULTADOS

Neste capitulo são abordados e discutidos os testes realizados para validação do modelo de classificação.

Para a validação do modelo foi utilizado o método de validação cruzada conforme mostrado no Código-fonte 8, obtendo-se uma precisão de 92,52% e um erro de 9,82%.

Código-fonte 8 – Validação cruzada

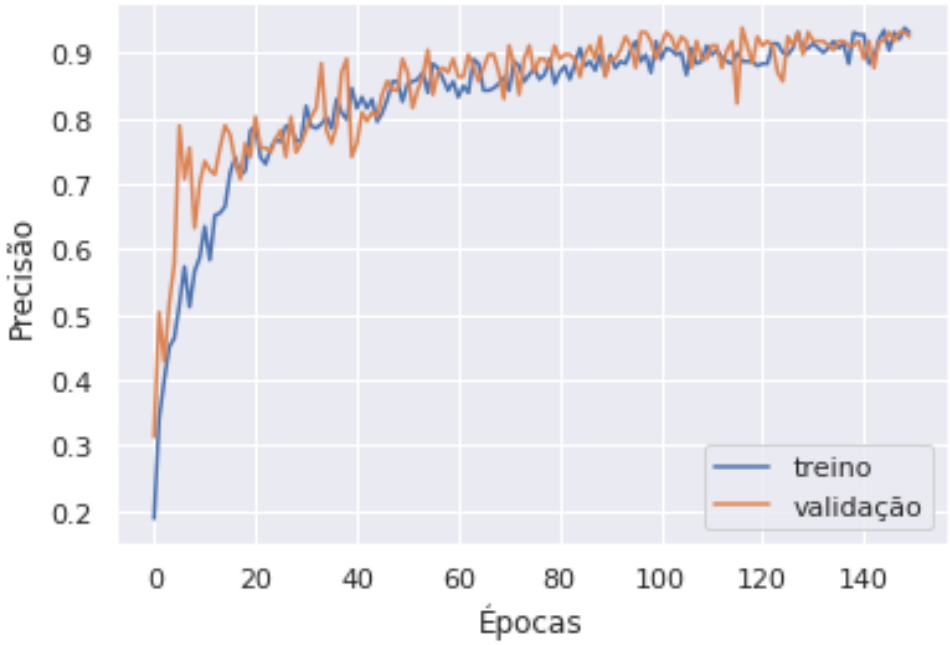
```
1 from sklearn.model_selection import cross_val_score
2 from sklearn import svm
3
4 scores = cross_val_score(model, X_test, y_test, cv=10)
```

Fonte: Autoria própria (2022).

Na Figura 18 podemos observar no eixo x, as épocas, e no eixo y, a precisão de acertos. Conforme o passar das épocas, a precisão tende a aumentar tanto para a base de treino quanto

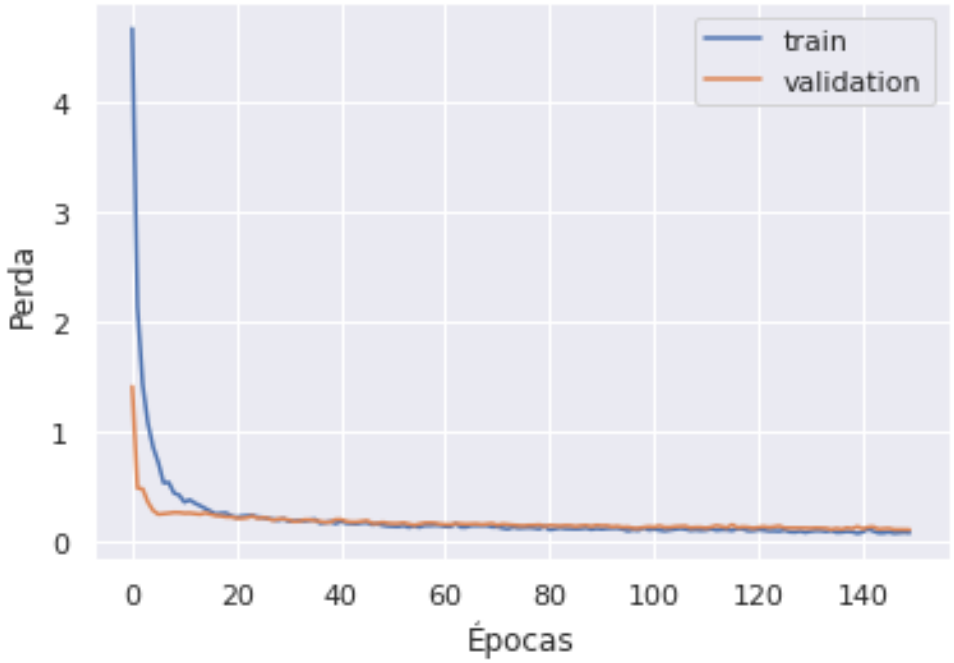
para a base de validação. Podemos observar pela Figura 19 que o erro começa com um valor alto e, conforme o passar das épocas, o erro diminui.

Figura 18 – Precisão



Fonte: Autoria própria (2022).

Figura 19 – Perda

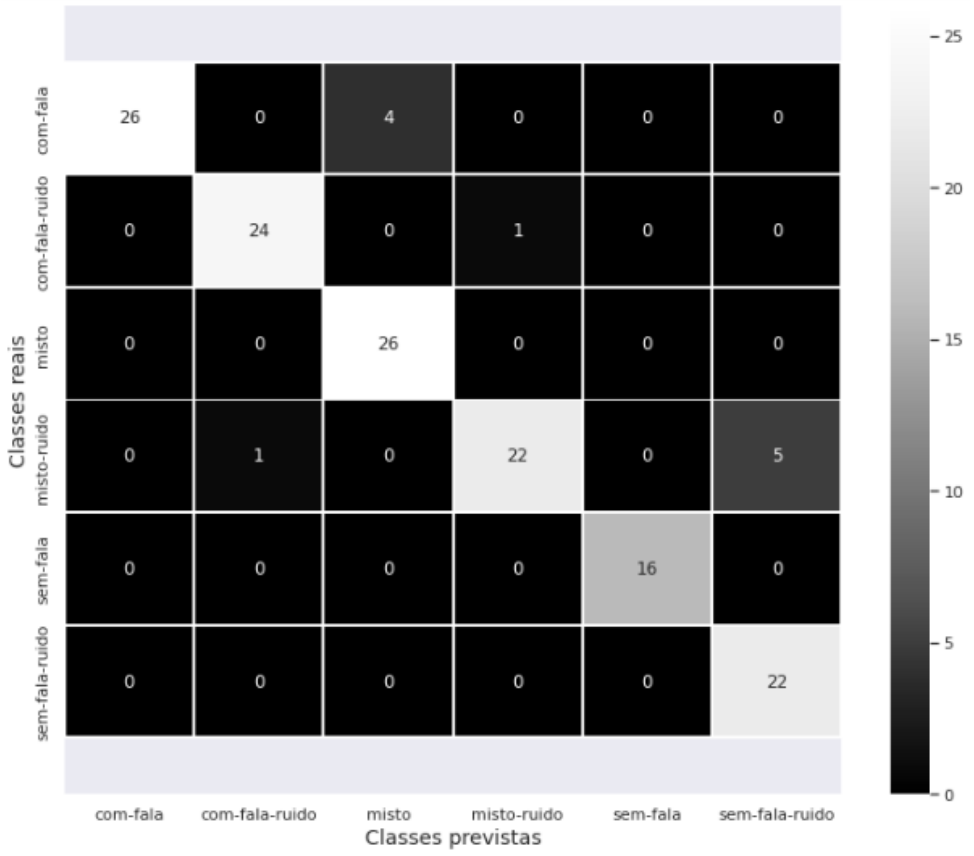


Fonte: Autoria própria (2022).

Com a matriz de confusão mostrada na Figura 20, podemos analisar detalhadamente a quantidade de acertos em cada classe. Com a matriz de confusão pode-se verificar as frequências

de classificação para cada classe do modelo, a diagonal principal contém a quantidade e acertos exatos de cada classe prevista.

Figura 20 – Matriz de confusão



Fonte: Autoria própria (2022).

As classe sem fala e sem fala com ruído são as principais a serem observadas, tendo em vista que são estes trechos de vídeo a serem descartados. A classe sem fala com ruido teve apenas uma classificação equivocada, a qual foi rotulado como misto com ruído. Já a classe sem fala obteve 100% de acerto. Contudo, a classe misto com ruído foi classifica diversas vezes como sendo sem fala e com ruído. Logo, esses trechos de vídeo seriam descartados erroneamente. O mesmo ocorre com a classe misto ao qual foi classificada uma vez como sendo sem fala. A classe que obteve menos acertos foi a mista com ruído, seguida da classe com fala.

Com a Tabela 1, podemos observar a taxa de acerto para cada uma das classes. A primeira coluna mostra a classe, a segunda coluna mostra a precisão da classificação, a coluna revocação define a porcentagem que o algoritmo consegue identificar determinada classe.

O melhor resultado de revocação são das classes sem fala, sem fala com ruído e misto, isso indica que o modelo consegue classificar corretamente 100% dos trechos de áudio que são desta classe e quando um áudio destas duas classes são detectados a classe sem fala teve uma

Tabela 1 – Relatório de classificação

Classe	**Precisão %**	**Revocação %**
com-fala	100	87
com-fala-ruido	96	96
misto	87	100
misto-ruido	96	79
sem-fala	100	100
sem-fala-ruido	81	100

Fonte: Autoria própria (2022).

precisão de 100% enquanto a classe sem fala com ruído obteve uma precisão de 81% e a classe misto teve 87% de precisão. O pior resultado vem da classe com misto com ruído, apresentando uma revocação de 79% com precisão de 96%.

4.3 AVALIAÇÃO DO MODELO

Para a realização da testagem foi enviado ao modelo trechos de vídeo inéditos pela rede neural. O vídeo original contém 1 minuto de 25 segundos, que foi dividido de forma aleatória em 24 trechos de vídeo contém minutagem diversa. Esses trechos foram nomeados de forma sequencial para realização da convergência final dos trechos classificados. Também, esses segmentos foram previamente classificados manualmente para verificar se a classificação foi realizada de forma correta após a passagem pelo modelo. A Tabela 2 mostra as classes previstas pelo modelo.

Como pode-se observar, o modelo obteve quatro classes previstas de forma incorreta, das quais são o trecho 7 que continha classe real como misto e classe prevista como sem fala. Dessa forma, esse trecho foi descartado erroneamente. Os trechos 13 e 14 contêm classe real com fala e ruído, porém foram classificados como com fala. Apesar do erro, estes trechos não foram

Tabela 2 – Classes previstas pelo modelo

Trecho	Classe Real	Classe Prevista	:	Trecho	Classe Real	Classe Prevista
1	sem-fala	sem-fala	:	13	com-fala-ruido	com-fala
2	sem-fala	sem-fala	:	14	com-fala-ruido	com-fala
3	com-fala	com-fala	:	15	sem-fala-ruido	sem-fala-ruido
4	misto	misto	:	16	sem-fala-ruido	sem-fala-ruido
5	sem-fala	sem-fala	:	17	com-fala-ruido	misto-ruido
6	com-fala	com-fala	:	18	misto-ruido	misto-ruido
7	misto	sem-fala	:	19	sem-fala	sem-fala
8	sem-fala	sem-fala	:	20	com-fala	com-fala
9	com-fala	com-fala	:	21	com-fala	com-fala
10	com-fala	com-fala	:	22	misto	misto
11	misto	misto	:	23	sem-fala	sem-fala
12	sem-fala-ruido	sem-fala-ruido	:	24	com-fala	com-fala

Fonte: Autoria própria (2022).

descartados do vídeo final editado. O mesmo acontece com o trecho 17, ao qual foi classificado como misto com ruído, apesar de ser um vídeo com fala e ruído.

5 DISCUSSÕES

Neste trabalho foi realizado o desenvolvimento de um modelo para classificação de trechos de vídeo. Este modelo foi treinado a partir de dados de vídeos de autoria própria desta forma os dados são apenas da voz de uma única pessoa. Entretanto, o modelo pode ser treinado com a voz de outras pessoas de forma separada e se obter o mesmo resultado, visto que o tom de voz humano em uma conversa tem escala parecida, independente do gênero, idade, entre outros. Contudo, no caso de dados de vídeos com vozes de várias pessoas ao mesmo tempo há a possibilidade do modelo obter menor precisão de acertos na classificação.

Os ruídos de fundo aumentam gradativamente de 20 em 20 porcento, sendo assim os trechos sem ruídos contemplam 0%, há trechos com 20% de ruído, 40%, 60%, 80% e por fim 100% onde a intensidade do ruído equivale à intensidade da voz. Desta forma, ruídos com maior intensidade podem vir a ser classificados de forma errônea com mais frequência comparado com os ruídos de menor intensidade. Também pode haver uma divergência de classificação com diferentes tipos de ruídos, um exemplo disso é de um vídeo ontem há uma televisão de fundo com pessoa falando, a voz vinda da TV pode ser confundida como uma voz válida.

Para fins de produtização, poderia ser implementado uma interface gráfica onde o usuário acrescenta como entrada seu vídeo bruto. Um algoritmo realiza todas as etapas de obtenção dos trechos de vídeos, entre elas os cortes aleatórios e a extração do áudio. Cada trecho é classificado pelo modelo. Por fim, os trechos classificados como sem fala e sem fala com ruído são descartados, o restante dos trechos são juntados novamente respeitando a linha do tempo do vídeo original, se obtendo como saída o vídeo editado.

6 CONCLUSÃO

Neste trabalho foi realizado o desenvolvimento de um modelo para classificação de trechos de vídeo. Para a obtenção dos trechos, os vídeos originais foram cortados utilizando um editor de vídeos. Foram elaborados dois códigos Python, um para a extração de áudio de cada trecho de vídeo e outro para todos os processos de criação do modelo, extração de características, implementação da rede neural convolucional, treinamento, testes e validação do modelo desenvolvido.

O modelo foi desenvolvido utilizando a técnica de MFCC para a extração de características de áudio obtendo-se espectrogramas equivalentes a imagens. A rede neural foi modelada utilizando a função de ativação relu para camadas ocultas, camadas de *dropout* e a função de ativação *softmax* para obter-se como saída probabilidades para cada classe. É realizado o treinamento da rede para por fim se alcançar um modelo de classificação de trechos de vídeo.

Foi possível atingir os objetivos propostos neste trabalho visto que o modelo desenvolvido é capaz de classificar a maioria dos trechos de forma assertiva, assim sendo possível se descartar trechos de áudio ao qual não contém falas, facilitando assim o trabalho de edição de vídeos. A depender da intensidade de ruídos de fundo, é possível que o modelo não tenha um resultado tão satisfatório, em sua maioria a classificação de trechos previamente como misto com ruído foram classificados como sendo sem fala com ruído. Assim, esses trechos poderão ser descartados erroneamente. A classe sem fala obteve um ótimo desempenho, podendo classificar corretamente 100% dos trechos de áudio com precisão de 100% na fase de validação.

Em sua maioria, a classificação de trechos previamente classificados como sem fala e sem fala com ruído obtiveram um obteve desempenho satisfatório. Entretanto, houveram alguns casos onde trechos com a classe mista com ruído foram classificados como sendo sem fala com ruído. Assim, esses trechos podem ser descartados erroneamente.

Para trabalhos futuros, propõe-se a exploração de diferentes ruídos em trechos de vídeos e aumento da base de dados para classificações mais assertiva. Também podem ser realizados testes no modelo com dados de voz de diferentes pessoas.

REFERÊNCIAS

BERRAR, D. **Cross-Validation.** [*S. l.: s. n.*], 2019.

CASARES, J. *et al.* Simplifying video editing using metadata. In: PROCEEDINGS of the 4th conference on Designing interactive systems: processes, practices, methods, and techniques. [*S. l.: s. n.*], 2002. P. 157–166.

CHAUHAN, R.; GHANSHALA, K. K.; JOSHI, R. Convolutional neural network (CNN) for image detection and recognition. In: IEEE. 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC). [*S. l.: s. n.*], 2018. P. 278–282.

CHEN, M. *et al.* Artificial neural networks-based machine learning for wireless networks: A tutorial. **IEEE Communications Surveys & Tutorials**, IEEE, v. 21, n. 4, p. 3039–3071, 2019.

CISCO. **Cisco Visual Networking Index Predicts Global Annual IP Traffic to Exceed Three Zettabytes by 2021**. [*S. l.: s. n.*], 2017. Disponível em: https://newsroom.cisco.com/press-release-content?type=webcontent&articleId=1853168. Acesso em: 07 outubro 2021.

DONG, S.; WANG, P.; ABBAS, K. A survey on deep learning and its applications. **Computer Science Review**, Elsevier, v. 40, p. 100379, 2021.

FACELI, K. *et al.* **Inteligência artificial: uma abordagem de aprendizado de máquina.** [*S. l.*]: LTC, 2011.

GAO, W. *et al.* Vlogging: A survey of videoblogging technology on the web. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 42, n. 4, p. 1–57, 2010.

HAQUE, A. Fft and wavelet-based feature extraction for acoustic audio classification. **International Journal of Advance Innovations, Thoughts & Ideas**, v. 1, n. 1, 2012.

HIBARE, R.; VIBHUTE, A. Feature extraction techniques in speech processing: a survey. **International Journal of Computer Applications**, Citeseer, v. 107, n. 5, 2014.

KIMBER, D.; WILCOX, L. *et al.* Acoustic segmentation for audio browsers. **Computing Science and Statistics**, Citeseer, p. 295–304, 1997.

LEAKE, M. *et al.* Computational video editing for dialogue-driven scenes. **ACM Trans. Graph.**, v. 36, n. 4, p. 130–1, 2017.

LIU, Z.; WANG, Y.; CHEN, T. Audio feature extraction and analysis for scene segmentation and classification. **Journal of VLSI signal processing systems for signal, image and video technology**, Springer, v. 20, n. 1, p. 61–79, 1998.

MINAMI, K. *et al.* Video handling with music and speech detection. **IEEE MultiMedia**, IEEE, v. 5, n. 3, p. 17–25, 1998.

NAIR, V. **The dummy's guide to MFCC**. [*S. l.: s. n.*], 2018. Disponível em: https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd. Acesso em: 12 novembro 2022.

O'SHEA, K.; NASH, R. An introduction to convolutional neural networks. **arXiv preprint arXiv:1511.08458**, 2015.

OKUN, J. A.; SUSAN ZWERMAN, V. *et al.* **The VES handbook of visual effects: industry standard VFX practices and procedures**. [*S. l.*]: Routledge, 2015.

OLIVA, J. T.; ROSA, J. L. G. Binary and multiclass classifiers based on multitaper spectral features for epilepsy detection. **Biomedical Signal Processing and Control**, Elsevier, v. 66, p. 102469, 2021.

OSISANWO, F. *et al.* Supervised machine learning algorithms: classification and comparison. **International Journal of Computer Trends and Technology (IJCTT)**, v. 48, n. 3, p. 128–138, 2017.

RODRIGUES, É. O.; CONCI, A.; LIATSIS, P. Morphological classifiers. **Pattern Recognition**, Elsevier, v. 84, p. 82–96, 2018.

TUN, P. T. Z. *et al.* Audio Feature Extraction using Mel-frequency Cepstral Coefficients. **International Journal Of All Research Writings**, v. 2, n. 12, p. 95–98, 2020.

UEDA, H. *et al.* Automatic structure visualization for video editing. In: PROCEEDINGS of the INTERACT'93 and CHI'93 Conference on Human Factors in Computing Systems. [*S. l.: s. n.*], 1993. P. 137–141.

VELAYUDHAM, V. **Audio Data Processing — Feature Extraction — Essential Science Concepts behind them — Part 2**. [*S. l.: s. n.*], 2020. Disponível em: https://medium.com/analytics-vidhya/audio-data-processing-feature-extraction-essential-science-concepts-behind-them-part-2-9c738e6a7f99. Acesso em: 11 novembro 2022.

WANG, T. *et al.* An evolutionary approach to automatic video editing. In: IEEE. 2009 Conference for Visual Media Production. [*S. l.: s. n.*], 2009. P. 127–134.

WARMBRODT, J.; SHENG, H.; HALL, R. Social network analysis of video bloggers' community. In: IEEE. PROCEEDINGS of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008). [*S. l.: s. n.*], 2008. P. 291–291.

ZHONG, D.; CHANG, S.-F. Structure analysis of sports video using domain models. In: IEEE COMPUTER SOCIETY. IEEE International Conference on Multimedia and Expo, 2001. ICME 2001. [*S. l.: s. n.*], 2001. P. 182–182.