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

Identificação de idioma em sinais de fala utilizando uma rede neural convolucional

Gabriel Fernandes

Santo André - SP
2021

Gabriel Fernandes

Identificação de idioma em sinais de fala utilizando uma rede neural convolucional

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

Universidade Federal do ABC – UFABC

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas

Programa de Graduação em Engenharia de Informação

Orientador: Kenji Nose Filho

Santo André - SP

2021

Gabriel Fernandes

Identificação de idioma em sinais de fala utilizando uma rede neural convolucional

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

Trabalho aprovado. Santo André - SP, 25 de novembro de 2021:

Kenji Nose Filho
Orientador

Professor
Ricardo Suyama

Mestre
Tito Caco Curimbaba Spadini

Santo André - SP
2021

Agradecimentos

O desenvolvimento deste Trabalho de Graduação contou com a ajuda, direta ou indiretamente, de diversas pessoas, dentre as quais agradeço:

Ao professor Kenji Nose Filho, que desde o primeiro momento de contato sobre o Trabalho de Graduação me acompanhou pontualmente, prestando todo auxílio necessário para a elaboração do projeto, além de explicar conceitos fundamentais para elaboração do presente trabalho.

Aos professores do curso de Engenharia de Informação, que através dos seus pacientes ensinamentos e participação ativa no processo de aprendizagem, permitiram que eu pudesse hoje estar concluindo este trabalho.

Aos meus pais, que me incentivaram a cada momento, me ofereceram todo o suporte para prosseguir os estudos e não permitiram que eu desistisse. E a minha avó Maria, que está sendo uma guerreira em um momento tão dificil da própria vida, e inevitavelmente me dá forças para superar qualquer desafio que me apareça.

Aos meus amigos, especialmente Juliana Oliveira Saldanha, Pietro Di Consolo Gregori e Vitor Miguel Martins, pelas discussões e intenso enriquecimento do meu progresso acadêmico, permitindo ouvir novos pontos de vista e aprender durante os anos de graduação.

Resumo

O presente trabalho propõe a identificação de idioma (alemão, espanhol e francês) em sinais de fala utilizando uma rede neural convolucional. O banco de dados utilizado foi o *Common Voice*, oferecido pela *Mozilla*. Para o processamento, os sinais de fala foram transformados em coeficientes mel-cepstrais (MFCC, do inglês *Mel-Frequency Cepstral Coefficients*) e, para a classificação, foi utilizada uma rede neural convolucional. De modo geral, o modelo gerado obteve bons resultados, obtendo uma acurácia média de **80%**, podendo ser utilizado em um sistema de reconhecimento de idioma em tempo real ou integrado a diversos outros sistemas de reconhecimento de fala. Além disso, os estudos poderiam ser ampliados para um futuro projeto de pós-graduação.

Palavras-chaves: voz, fala, classificação, áudios, idiomas, MFCC, redes neurais convolucionais.

Abstract

The present work proposes the identification of language (German, Spanish and French) in speech signals using a convolutional neural network. The database used was the *Common Voice*, offered by *Mozilla*. For the processing, the speech signals were transformed into Mel-Cepstral Coefficients (MFCC) and, for the classification, a convolutional neural network was used. In general, the generated model obtained good results, reaching an average accuracy of **80%**, and could be used in a real-time language recognition system and/or integrated into several other speech recognition systems. Furthermore, the studies could be expanded for a future postgraduate project.

Keywords: voice, speech, classification, audios, languages, MFCC, convolutional neural networks.

Lista de ilustrações

Sumário

1 Introdução

Problemas envolvendo a análise de dados de entrada para predizer padrões como conteúdo, idioma, gênero, identidade e diversas outras informações são bastante recorrentes e tem sido desenvolvidos cada vez mais com o avanço de ferramentas computacionais capazes de analisar grandes quantidades de dados e gerar modelos mais complexos, rápidos e assertivos. Dentre eles podemos destacar alguns (LAZZARI, 1999):

- *Automatic Speaker Identification*: Capacidade da máquina identificar corretamente a identidade do autor da gravação de um áudio dentre um conjunto finito de possíveis autores, utilizado, principalmente, em temas como segurança digital e privacidade.

  Um dos exemplos de aplicações práticas mais presentes no dia a dia que envolve esse reconhecimento é o *"Ok Google"*, como ilustrado na Figura 1, presente nos celulares com sistema operacional *Android* que, a partir de um treinamento realizado com poucas amostras de voz do próprio usuário, habilita a Assistente de Voz do *Android*.

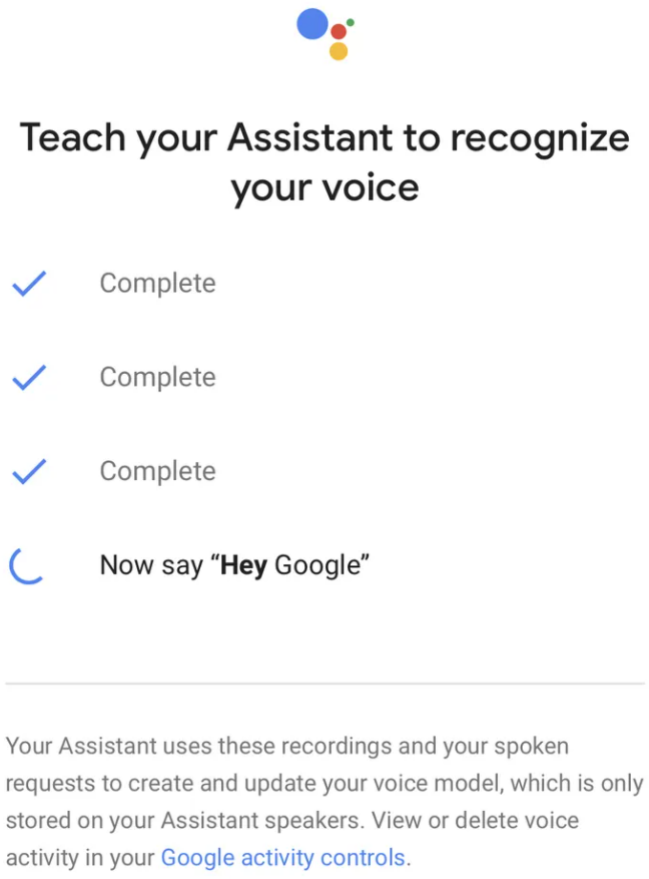

Figura 1 – Exemplo da etapa de configuração de voz do *"Ok Google"* (Lauren Barack, GearBrain, 2019).

- *Automatic Spoken Language Identification*: Capacidade da máquina identificar corretamente o idioma falado por um indivíduo. Para problemas computacionais é necessário transformar o áudio de entrada separando-o em bancos de filtros específicos que sejam capazes de evidenciar parâmetros intrínsecos da fala, como o tom, as frequência das formantes e a forma geral de onda do sinal, similar ao processamento natural feito pelos canais auditivos. Após o processamento inicial é preciso também gerar modelos de classificação que sejam eficientes e eficazes para classificar os diferentes idiomas a partir de suas peculiaridades.

  Em comparação com o computador é válido dizer que o humano é a melhor 'máquina' capaz de realizar essa tarefa, fazendo analogias com referências guardadas em seu subconsciente ou em experiências passadas, o ser humano é capaz de predizer, com certa confiabilidade, qual o idioma falado por algum individuo, mesmo que não seja fluente ou tenha grande quantidade de conhecimento sobre o idioma.

- *Automatic Text Language Identification*: Capacidade de identificar qual idioma um determinado texto está escrito. Graças ao avanço computacional, é possível comparar grandes arquivos de texto com palavras presentes nos dicionários de grande parte das línguas já catalogadas e, assim, predizer com uma acurácia próxima a 100% o idioma em que o texto foi escrito. Logo, é um processo bem definido.

- *Spoken Language Translation*: Capacidade da máquina em realizar traduções a partir de entradas de áudio. Atualmente são encontradas diversas soluções para esse problema em equipamentos dos mais diversos nichos, como Assistentes de Voz (*Alexa*, *Siri*, *Cortana*, etc.), *gadgets* que analisam áudios em tempo real e fornecem traduções, como por exemplo o *WT2 Plus AI Realtime Translator Earbuds* (Timekettle, 2021), e tambem o Google Tradutor, que, a partir de uma entrada de áudio, prediz com alta acurácia qual é o idioma e oferece uma sugestão de tradução instantaneamente.

Além dos problemas acima citados, um outro problema bastante interessante envolvendo o processamento de sinais de áudio e a programação de linguagem natural é a conversão de texto para fala, conhecido como *Text to Speech*, em que podemos destacar o *Watson Text to Speech*, desenvolvido pela IBM (IBM, 2021) e o *Transcribe*, ilustrado na Figura 2, desenvolvido pela Amazon (Amazon, 2021b). Outra ferramenta da Amazon é a atuação da Alexa focada em soluções de negócio, que utiliza o mesmo mecanismo da versão comercial da Alexa, porém totalmente programável e adequável ao ambiente único de cada corporação (Amazon, 2021a).

Para este trabalho a ideia é fornecer uma solução de *Automatic Spoken Language Identification* que seja capaz de identificar qual idioma está sendo falado dentro do seguinte grupo finito de idiomas: Alemão, Espanhol e Francês. Para isso, foram utilizados dados extraídos do banco de dados gratuito do programa colaborativo *Common Voice* da *Mozilla*

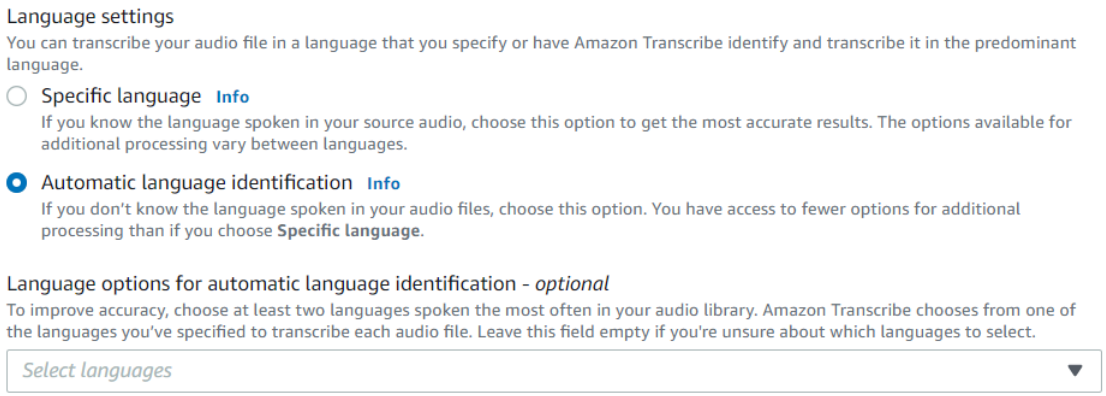

Figura 2 – Exemplo da ferramenta *Amazon Transcribe* durante a configuração de detecção de idioma de forma automática. (Amazon, AWS Official Documentation, 2021)

(MOZILLA, 2021), em que é possível participar ativamente da criação do banco de dados através da gravação de áudios ou até da classificação da qualidade dos áudios de outros usuários.

Soluções parecidas já são encontradas dentro de ferramentas mencionadas como o Assistente de Voz da *Google*, a *Alexa* e o *IBM Watson*, porém, como ferramenta de suporte, considerando que essas ferramentas possuem outros objetivos. Além de identificar o idioma falado, estas ferramentas buscam também realizar a sua tradução e interpretação, para incluir diversas funções de pós-processamento e integração a outros sistemas. Inclusive dentro da ferramenta *Amazon Transcribe* é possível observar a acurácia de idioma que o modelo previu a partir da entrada de áudio (Julien Simon, AWS News Blog, 2021), resultado semelhante ao que deseja-se obter neste trabalho.

Trabalhos e artigos similares foram pesquisados para servir de inspiração e referência para a criação deste trabalho, entre eles encontra-se um trabalho de reconhecimento de identidade de indivíduo (*Automatic Speaker Identification*). A abordagem utilizada como pré-processamento foi o cálculo de MFCC, junto com histogramas de frequências de tom, e a detecção de gírias e de frequências de tons. Já a parte de processamento foi realizada pelo algoritmo de *Harmonic Product Spectrum* (HPS). Foram utilizados áudios de idiomas Inglês, Marathi e Hindu, com resultados de identificação entre 81 e 87% (CHOUGULE; REGE, 2007).

Em outro trabalho utiliza-se uma rede neural artificial para identificação de identidade de indivíduo em mais de um idioma, coletando vozes de 19 pessoas distintas, masculinas e femininas, para 8 idiomas: Catalão, Francês, Finlandês, Italiano, Português, Indonésio, Hindu e Inglês. O treinamento dos dados é realizado utilizando o algoritmo de *Back Propagation* (BP), algoritmo iterativo projetado para minimizar o erro quadrático médio entre a saída real de um perceptron multi-camadas e a saída desejada. O trabalho apresenta acurácia média próximo a 96% (AGRAWAL; KAUR; KAUR, 2012).

Direcionado a identificação de idiomas, o trabalho escrito por (KUMAR et al., 2004) faz uso de cadeias de Markov e análise de sons de consoantes e vogais entre dois idiomas próximos entre si: Tamil e Hindi. A ideia é predizer qual o idioma do áudio baseado em características acústicas. Os resultados de acurácia obtidos são próximos a 1, o que são excelentes e utilizam técnicas adaptativas para discursos multi-idiomas.

Utilizando um método um pouco diferente, (LI; LEE, 2007) tenta fazer a predição de idioma falado a partir do uso de redes neurais em VSM (*Vector Space Modeling*), de acordo com os resultados exibidos os valores chegam próximos a 98% e 96% quando variado a base de dados entre diferentes versões da NIST LRE (*Language Recognition Tasks*), 1993 e 2003. Outra tentativa de método para predição de idioma falado é feito em (GELLY; GAUVAIN, 2017) onde é utilizado aproximações angulares baseadas em LSTM (*Long Short Term Memory*), junto com redes neurais regressivas obteve-se resultados de acurácia abaixo de 80%.

A análise a ser realizada neste trabalho irá envolver o pré-processamento demonstrado em (CHOUGULE; REGE, 2007) através de MFCCs e bancos de filtros, aplicados a redes neurais artificiais, mais especificamente, redes neurais convolucionais. Essa escolha é feita considerando a presença de mais de uma dimensão nos dados, por conta da MFCC, e a capacidade de processamento disponível.

Uma das motivações e inspirações para essa implementação foi o trabalho (Fernandez, Lucero Guadalupe, 2020), proposta presente em um desafio na plataforma *Kaggle*, em que a autora desenha uma rede neural convolucional para classificar idiomas utilizando áudios ".flac" e um método próprio de calcular MFCC, os resultados apontam acurácia média de 99% para validação e 97% para teste.

Em relação a quantidade de áudios, foram extraídos todos os áudios da versão "*Common Voice Corpus 6.1*" para os idiomas Alemão, Espanhol e Francês. Para cada idioma, o banco de dados é composto de aproximadamente 250 mil amostras (aproximadamente 17GB/idioma). Para este trabalho, apenas uma parcela dos dados foi utilizada, cerca de 39 mil amostras inicialmente, dividida em conjunto de treinamento, validação e teste.

O trabalho foi dividido da seguinte forma: Fundamentação teórica, onde são descritos conceitos teóricos da solução criada além da motivação de utilizá-las e seu papel durante o produto. Em seguida, na seção de Implementação realizada, é descrita a implementação com detalhes do processamento dos dados, programação e geração do modelo da rede neural. Para a seção de Resultados são detalhados os resultados obtidos na validação e teste do banco de dados "*Common Voice Corpus 6.1*" além de testes realizados com dados retirados de vídeos do *Youtube*, discutidos e analisados. E, por último, é feita uma conclusão dos resultados da solução apresentada e são sugeridas ideias de melhorias para passos futuros.

2 Fundamentação teórica

Como mencionado no capítulo 1, a intenção deste trabalho é realizar a classificação baseada em padrões de fala característicos de cada idioma e, por meio de uma rede neural convolucional, obter um modelo que seja capaz de determinar o idioma falado a partir de uma entrada de áudio. Para isso é necessário compreender alguns aspectos e características dos sinais de fala, que serão abordados a seguir.

2.1 Características dos Sinais de Áudio

Áudios podem ser caracterizados como qualquer som ou ruído que os ouvidos humanos possam reconhecer, também considerados como uma das possíveis fontes de sinais analógicos, áudios usualmente são digitalizados por meio de diversos métodos para serem reconhecidos por dispositivos digitais e, assim, reproduzidos o mais fielmente possível.

Um sinal de áudio digitalizado possui parâmetros como a **taxa de amostragem**, que representa a quantidade de amostras por segundo (unidade em Hertz) retirada da fonte daquele sinal, e também a **resolução**, normalmente representada em número de bits, que representa a quantidade de bits que foram utilizados para guardar cada amostra daquele sinal. Exemplo, para CDs estéreos são 44100 Hz e 32 bits de resolução, enquanto para MP3 pode variar entre 8000 Hz e 48000 Hz para taxa de amostragem e 32 bits de resolução, no entanto para MP3 é comumente chamado de taxa de bits, e este valor é normalmente entre 16 kb/s e 320 kb/s (Wikipedia contributors, 2021). É importante ressaltar que o MP3 realiza uma compressão com perdas de componentes de alta frequência, menos percebidas para seres humanos, enquanto pode ser notada por computadores e sistemas que analisam a frequência do sinal, assim é um ponto a se ter em mente para sistemas que consideram áudios em formato MP3 como sinal de entrada.

2.2 Características dos Sinais de Fala

Os elementos de comunicação de um indivíduo o caracterizam e o distinguem de outros individuos, sendo a sua fala um desses elementos. Para o sinal de fala, a intensidade (ou amplitude), o sotaque, os intervalos entre palavras e a escolha de palavras diferem entre pessoas e ambientes, e até são adaptados durante as diferentes etapas de vida; porém, alguns padrões são possíveis de serem observados entre individuos de mesma nacionalidade, sexo, cultura, etc. Essas características são utilizadas como forma de identificação para tecnologias de reconhecimento de fala ou até mesmo no cotidiano, onde o seu sotaque, a

sua escolha de palavras ou até o seu tom de voz podem identificá-lo ou identificar traços de sua origem.

Para análise de sinais de fala é preciso inicialmente fazer a distinção entre sinais vozeados e sinais não vozeados: sinais vozeados são definidos por grande concentração de energia em baixas frequências, já sinais não vozeados se assemelham a ruídos, possuem energia distribuída entre intervalos maiores de frequência. Um exemplo prático e simples para compreender essa diferença é a pronúncia da letra /a/ e da letra /x/, ao falar a letra /x/ temos a vibração das cordas vocais caracterizando um sinal vozeado, já para letra /x/ não conseguimos distinguir a vibração das cordas vocais, e ao analisar o espectro de frequência, assemelha-se a ruído; portanto, é um sinal não vozeado.

Quando olhamos um sinal vozeado de fala, objeto do nosso estudo pelas características de espectro, analisamos uma janela de sinal no domínio da frequência, separando em intervalos de aproximadamente 20 ms para sinais de fala e, com isso, é possível identificar picos de amplitude sequenciais, chamados de **formantes** F_i (Figura 3). Usualmente apenas os três primeiros picos contribuem mais para a inteligibilidade da fala. O pico F_1 concentra maior parte da energia do sinal, enquanto os picos F_2 e F_3 contribuem com informações de fala.

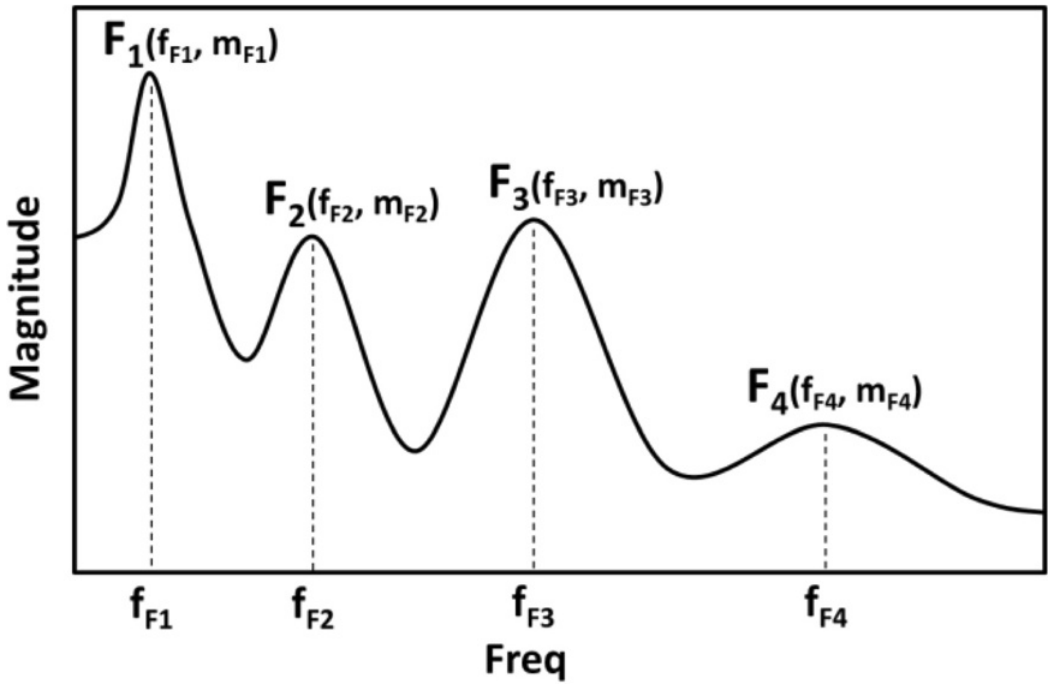

Figura 3 – Exemplo de sinal de fala com identificação das formantes. (SHIN; CHO, 2014)

A partir da janela de fala também é possível realizar a classificação dos sons emitidos, em decibéis, por exemplo, geralmente, sons de vogais ocupam o intervalo entre 21 dB e 31 dB, já sons anasalados ocupam entre 14 dB e 20 dB, etc. Porém, mesmo que a amplitude de vogais sejam maiores que a amplitude de consoantes, a inteligibilidade de

consoantes acabam sendo maiores que as de vogais, pela diferença das demais formantes (MCLOUGHLIN, 2009).

No domínio da frequência, a primeira formante (F_1) está localizada próximo a 500 Hz para homens e 800 Hz para mulheres, e, como dito anteriormente, representa a região do espectro onde está concentrada a maior parte da energia. Já as formantes F_2 e F_3 ficam localizadas entre 800 Hz e 3 kHz, carregando informações do conteúdo da fala.

Para demonstrar essa afirmação observa-se a Figura 4, onde os dados de entrada são silabas de fala, e foram aplicados os dois filtros da imagem, um filtro passa-altas a partir de 2 kHz e um filtro passa-baixas em 1 kHz. O eixo y com as barras verticais representa a "articulação", ou seja, a inteligibilidade da fala. Analisando os resultados do gráficos temos que, no caso do filtro passa-baixas (LP) aproximadamente 25% do conteúdo da fala consegue ser recuperado, enquanto para o filtro passa-altas (HP) o valor é cerca de 70%.

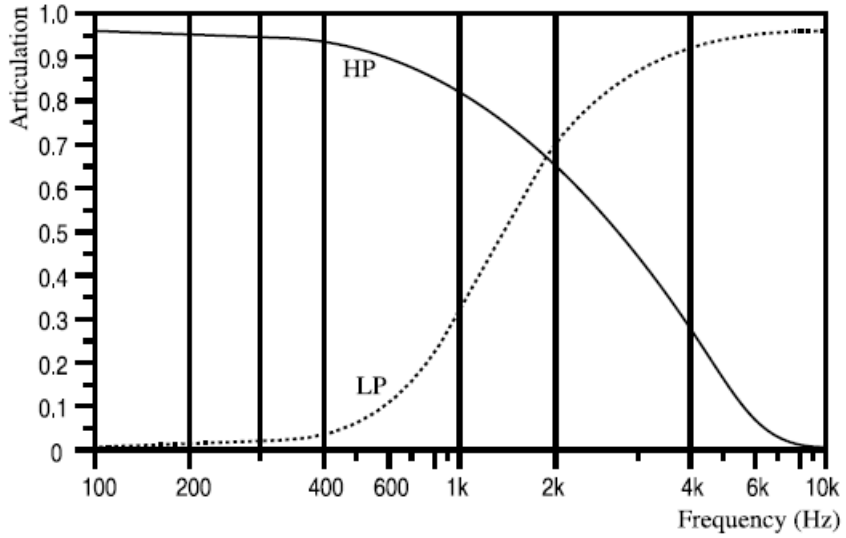

Figura 4 – Exemplo de filtros passa-altas (HP) e passa-baixas (LP) em um sinal de fala. (MCLOUGHLIN, 2009)

Além dos parâmetros teóricos de fala explicados podemos também acrescentar parâmetros referentes ao conteúdo de cada áudio, como a entonação, fisicamente representados pelas frequências de vibração do áudio, o tom, que denota a taxa de vibrações de áudio. Assim, os tons altos e baixos são percebidos de forma diferente pelo aparelho auditivo. Todos esses parâmetros serão imprescindíveis na análise de áudios porque estão intrínsecamente ligados às características inatas de qualquer idioma, diretamente ou indiretamente.

2.3 MFCC

Para extrair essas características dos áudios utiliza-se diversos algoritmos e análises, tanto no domínio do tempo quanto no domínio da frequência. Além disso são utilizados os conceitos abordados de sinal de fala para separar os áudios em janelas, aproveitando o intervalo das principais formantes. Um dos métodos que iremos utilizar nesse produto é chamado de *Mel Frequency Cepstral Coefficient*, MFCC, um método computacional criado para tentar simular o funcionamento do aparelho auditivo humano (CHAKROBORTY; ROY; SAHA, 2006), que seleciona janelas dentro do espectro de frequência do áudio com diferentes pesos, maximizando a influência de sinais de baixa frequência em relação a sinais de alta frequência, por esse motivo aproxima-se do corpo humano, com influência linear para componentes de baixa frequência e logarítmica para componentes de alta frequência.

Resumidamente o algoritmo de MFCC funciona da seguinte forma: o sinal é dividido em uma quantidade pré-definida de janelas de análise; em seguida, para cada janela é calculada a densidade de potência estimada, simulando a cóclea humana. Ao resultado do passo anterior, também utilizando como base o funcionamento da cóclea, aplicam-se os bancos de filtro em frequência *Mel* e somam-se as energias das janelas em cada filtro, que não reconhece individualmente duas componentes próximas em frequência. Calcula-se o logaritmo dos bancos de filtro, porque o sistema auditivo humano não ouve em escala linear e, em seguida, calcula-se a *Discrete Cosine Transform*, DCT, para descorrelacionar as energias dos bancos de filtro na escala logarítmica; por último, descarta-se metade das DCTs calculadas devido à sobreposição de energia resultante da operação (CRIPTOGRAPHY, 2013).

Em seguida detalha-se o cálculo do banco de filtros da MFCC seguindo o algoritmo (LI; COX, 2019), com fluxograma descrito na Figura 5, e o resultado visual do processo pode ser conferido na Figura 6:

i. Definir os parâmetros de janelas que serão criadas na extensão do espectro do sinal: tamanho, duração e número de janelas;

ii. Para cada janela é preciso calcular a *Fast Fourier Transform*, FFT, e pegar seu módulo representando a densidade de potência estimada da janela;

iii. Substituir as frequências de áudio em cada janela e aplicar a fórmula de conversão de Hertz para a escala Mel para encontrar os centros da escala Mel:

$$mel(f) = 1127.0148\ log\left(1 + \left(\frac{f(Hz)}{700}\right)\right) \tag{2.1}$$

iv. Converter os centros encontrados para Hertz;

v. Encontrar os filtros triangulares para os centros convertidos e calcular o logaritmo da respectiva energia seguindo a fórmula:

$$Log\ Energy = log \sum_{n=1}^{W} S_n^2 \tag{2.2}$$

Onde W é o tamanho da janela;

vi. Por último é necessário calcular a DCT do vetor de energia para descorrelacionar os filtros calculados, desconsiderando metade dos valores por conta do *overlap* de energia resultante da operação.

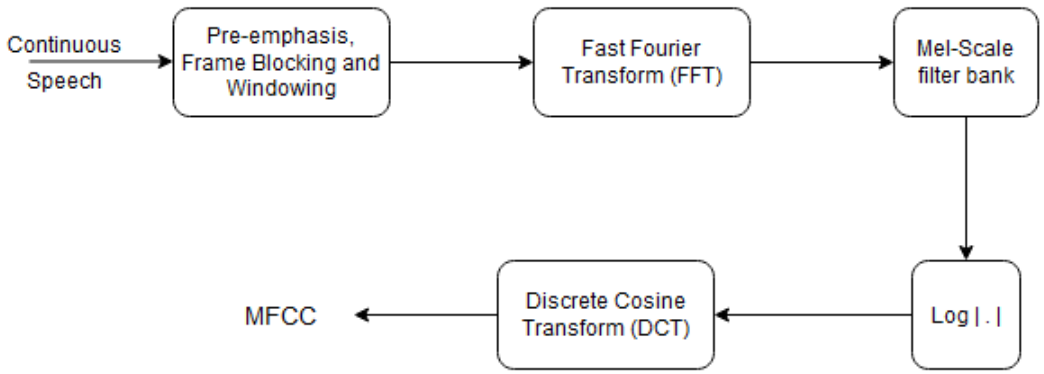

Figura 5 – Fluxograma do cálculo da MFCC para uma entrada contínua de áudio (ALIM; RASHID, 2018).

O processo de cálculo da MFCC deste produto foi realizado seguindo a biblioteca de tratamento de áudio e linguagem *python-speech-features*. (Lyons (2013))

2.4 Redes Neurais Artificiais

De acordo com (HAYKIN, 2009), redes neurais artificiais são estruturas em rede que tentam simular sistemas neurais humanos a partir da interligação de neurônios, assim sendo, neurônios são a estrutura fundamental de uma rede neural. Para redes neurais aplicadas a problemas de *Machine Learning*, o foco está em criar redes de neurônios com parâmetros ajustáveis que realizam operações matemáticas com o vetor de entrada para resultar em um vetor de saída que otimize métricas de saída.

A ideia principal é que a rede auto ajuste os parâmetros dos neurônios a partir de uma função de suporte, chamada de *bias*, durante rodadas de execução, chamadas de épocas, assim, otimizando o resultado do vetor de saída a partir do vetor de entrada. Um exemplo de rede neural artificial está ilustrado na Figura 7.

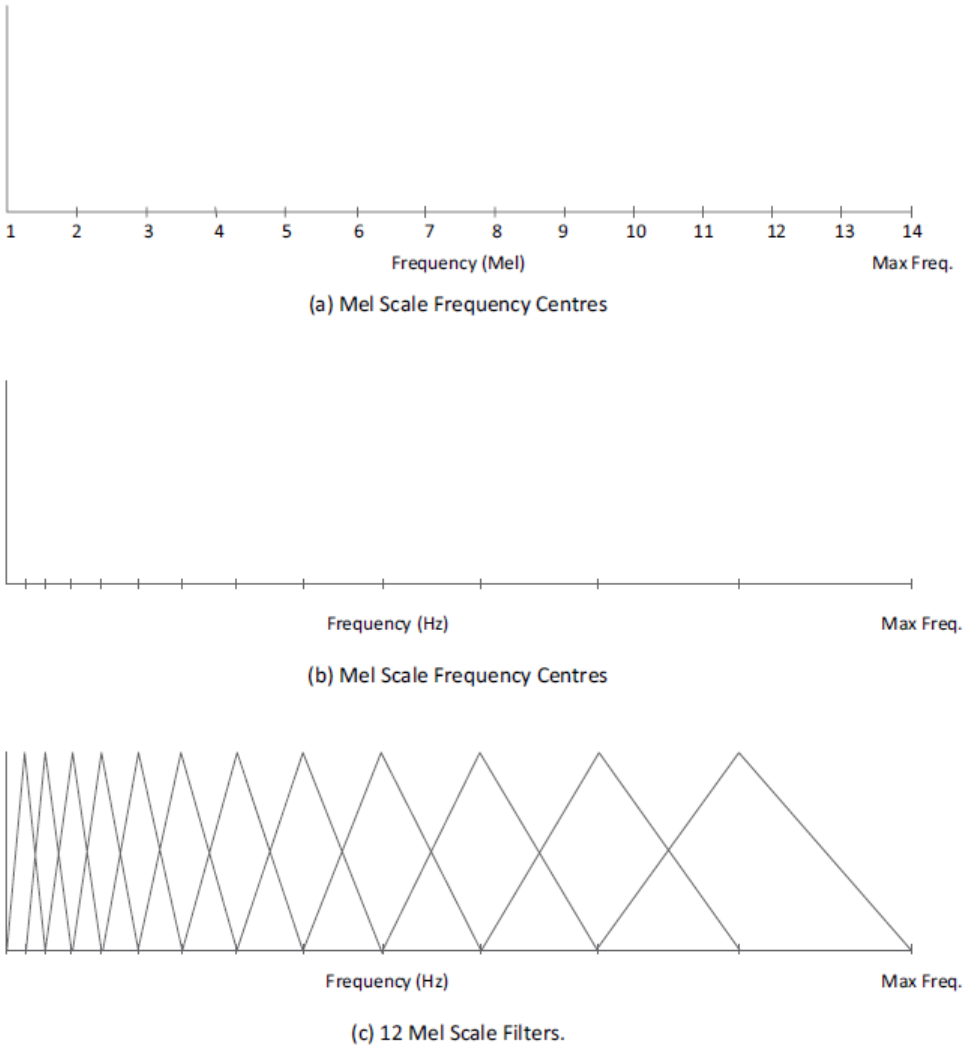

Figura 6 – Visualização dos passos iii. e iv. do algoritmo de cálculo da MFCC (LI; COX, 2019).

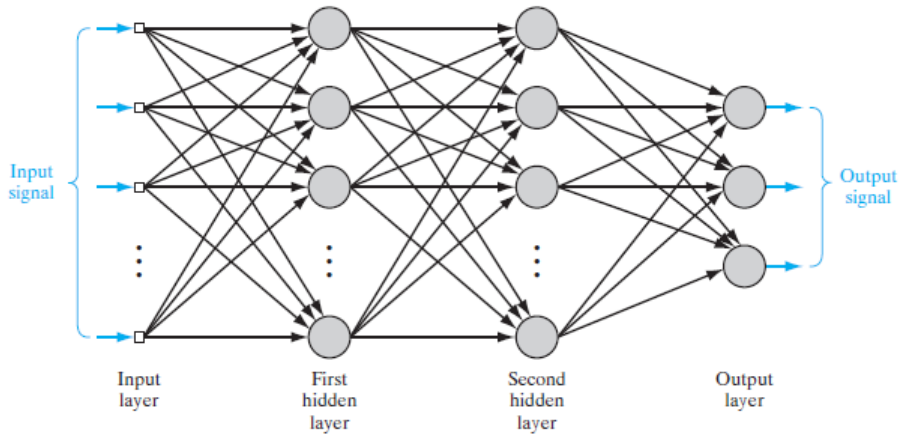

Figura 7 – Exemplo de rede neural artificial com multiplas camadas de neurônios. (HAYKIN, 2009)

Outros motivos de uso de redes neurais é a plasticidade com que é possível treinar os coeficientes a partir dos dados de entrada, ou seja, a possibilidade que os neurônios têm de se adaptarem a vetores de entrada para gerarem saídas cada vez melhores, e assim moldar a rede neural com os resultados desejados.

Redes neurais também podem ser não-lineares, e além de acarretar em maior complexidade, torna a relação de coeficientes mais precisa porque não tenta encaixar os dados em padrões lineares de dispersão. Também são tolerantes a falhas, ou seja, por mais que uma rede neural possa cometer falhas, ela se auto ajusta para aproveitar as falhas e melhorar na próxima iteração, ajustando seus coeficientes internos com o objetivo de maximizar as métricas de desempenho do modelo.

Um exemplo de rede neural artificial, que será utilizada nesse estudo, é a rede neural convolucional, que trabalha com diversas camadas e cálculos de convoluções para gerar um vetor classificador, e assim predizer um vetor de entrada.

2.5 Redes Neurais Convolucionais

Redes convolucionais são classes de redes neurais voltadas para dados multidimensionais com alto grau de invariância a translações, escalamentos e distorções no geral. Essa rede é baseada nos princípios de: (Saha, Sumit, 2018)

1. *Convolution*: São definidos dimensões de cada camada da rede e aplicado o conceito de convolução, ou seja, a integral do quanto uma função $f(x)$ percorre sobre uma função $g(x)$, considerando que $g(x)$ permanece imóvel.

   A convolução em vetores discretos atua como um mapa de features, reduzindo vetores com muitas dimensões para vetores menores, preservando a informação principal; (Eric W., Weisstein, 2021)

2. *Pooling*: Técnica utilizada para diminuir o tamanho do mapa de *features* gerado no processo de convolução, tanto por motivos de processamento quanto por motivos de redução de complexidade;

3. *Padding*: Representa a técnica a ser utilizada na convolução, sendo possível gerar mapas de *features* de mesmo tamanho (*same*) ou menores (*valid*), com o intuito de diminuir sua complexidade. No nosso caso utilizamos ambas técnicas: inicialmente o *padding valid* para diminuir o tamanho de dimensões do vetor e em seguida o *padding same* para manter o tamanho total e trabalhar nas camadas de convolução;

4. *Classification*: Último processo da rede neural convolucional. O princípio de classificação gera um vetor unidimensional como saída para atuar como o vetor de pesos, que será utilizado para classificar os dados de entrada em dados de saída, além disso um

de seus parâmetros de ativação chamado *Softmax*, atua aprendendo as componentes mais importantes dos dados de entrada e priorizando a adequação delas para o dado de saída, assim, ajustando a rede para trabalhar melhor com pouca variância entre si.

Um exemplo de rede neural convolucional está demonstrado na Figura 8, onde o dado de entrada é a foto de um koala. São utilizadas duas camadas de *convolution* + *pooling* para extração de *features* importantes que caracterizam a foto; em seguida, é determinado um vetor de classificação com pesos distintos e, ao final da análise, é classificado como um koala (Baheti, Pragati, 2021).

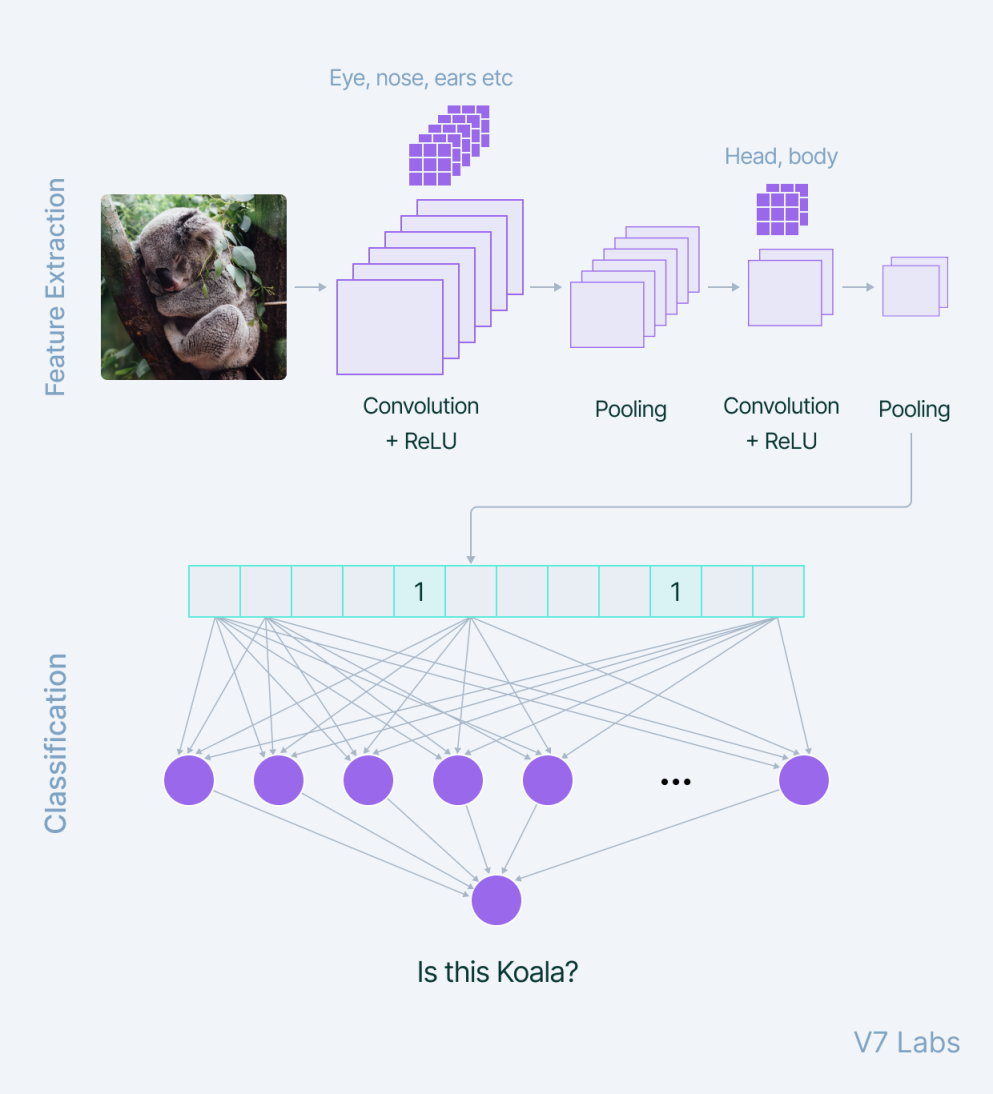

Figura 8 – Exemplo de rede neural convolucional utilizada para reconhecimento de uma imagem de koala. (Baheti, Pragati, 2021)

Redes neurais convolucionais recebem nomes únicos pelos seus autores para representarem configurações de camadas, convoluções, *pooling* e *padding* que melhor se adequam para desempenhar determinado papel na solução de um problema. Algumas das principais

redes convolucionais são: *LeNet*, *AlexNet*, *VGGNet*, *GoogLeNet*, *ResNet*, *DenseNet*, *ZFNet*, etc (Baheti, Pragati, 2021).

Por exemplo a rede neural convolucional *LeNet* foi a primeira rede convolucional criada. *LeNet* foi treinada em imagens 2D em escala de cinza com tamanho 32x32x1. Seu principal objetivo era gerar um modelo de identificação de imagens de assinaturas feitas a mão em cheques bancários. Sua configuração espacial eram duas camadas de *convolution* + *pooling* conectadas por duas camadas para classificação.

2.6 Métricas de Análise de Desempenho

Para avaliar os resultados obtidos, iremos utilizar as seguintes métricas de avaliação de desempenho:

- Matriz de confusão: Matriz C que reune a precisão da classificação do modelo, onde $C_{i,j}$ representa o elemento que foi predito para o grupo i, sendo que pertence ao grupo j. Ou seja, a contagem de *verdadeiros positivos* (TP, *True Positives*) é $C_{0,0}$, de *falsos positivos* (FP, *False Positives*) é $C_{1,0}$, de *verdadeiros negativos* (TN, *True Negatives*) é $C_{1,1}$ e de *falsos negativos* (FN, *False Negatives*) é $C_{0,1}$.

  Em suma, é valido dizer que, os elementos da matriz de confusão são definidos da seguinte forma:

  - a_{ii} - Elemento predito corretamente como sendo do idioma i considerando que seu idioma é de fato i;
  - a_{ij} - Elemento predito incorretamente como sendo do idioma i considerando que seu idioma na verdade é j.

  A Figura 9 representa a interpretação dos resultados baseado na localização deles na matriz de confusão Scikit-learn (2007-2020).

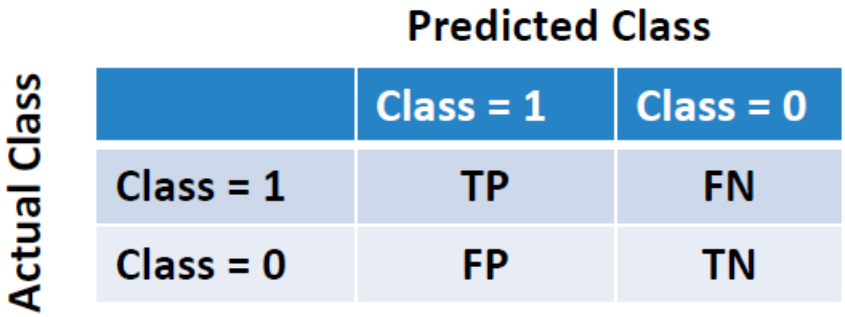

Figura 9 – Generalização da matriz de confusão baseado no resultado conhecido e no resultado predito.

- *Acurácia* (*Accuracy*): Representa a porcentagem dos dados que foram corretamente classificados.

$$Acurácia = \frac{TP + TN}{TP + TN + FN + FP} \tag{2.3}$$

- *Taxa de erro* (*Error Rate*): Representa a porcentagem dos dados que não foram corretamente classificados, ou seja, é o complemento da acurácia.

$$Taxa\ de\ erro = \frac{FP + FN}{TP + TN + FN + FP} \tag{2.4}$$

- *Precisão* (*Precision*): Representa a porcentagem dos dados classificados como positivo que realmente são dados positivos.

$$Precisão = \frac{TP}{TP + FP} \tag{2.5}$$

- *Revocação* (*Recall*): Representa a porcentagem dos dados classificados como positivo que o modelo calculou corretamente como positivos.

$$Revocação = \frac{TP}{TP + FN} \tag{2.6}$$

- *F-Score*: Representa a média harmônica entre a precisão e o recall, descritos anteriormente.

$$F\text{-}Score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{2.7}$$

Estas métricas são consideradas padrões para problemas de classificação e serão utilizadas em todos os testes que serão realizados com diferentes modelos. Nesta solução busca-se maximizar, principalmente, a acurácia do modelo, ou seja, maximizar a porcentagem de dados classificados corretamente em relação ao idioma do conteúdo do áudio correspondente.

3 Implementação realizada

A implementação realizada foi dividida em quatro etapas para auxiliar no caminho lógico desenvolvido: pré-processamento dos dados, criação da rede neural convolucional, análise das métricas de desempenho e resultados adicionais.

3.1 Pré-Processamento dos Dados

Conforme mencionado anteriormente os dados foram retirados da base de dados livre disponibilizada pela empresa *Mozilla*, "Common Voice Corpus 6.1". Os idiomas utilizados são alemão, francês e espanhol. Ao realizar o download da base de dados completa, percebe-se que os dados de áudio estão reunidos em formato ".mp3" em uma única pasta, enquanto tem-se diversos arquivos ".tsv", entre eles de teste, treino, desenvolvedores, dados inválidos, etc. Todos eles contendo cabeçalhos dos dados de áudio para análise em modelos de *Machine Learning* e *Data Science* (MOZILLA, 2021).

Inicialmente foi preciso compreender como estavam estruturados os arquivos de cabeçalho do banco de dados, quais informações que continham, qual era a organização e o conteúdo de cada coluna, além de verificar a qualidade e a clareza dos sinais de áudio. Para isso, são representados nas Figuras 10 e 11 os sinais nos domínios da frequência e do tempo de um áudio de exemplo, respectivamente.

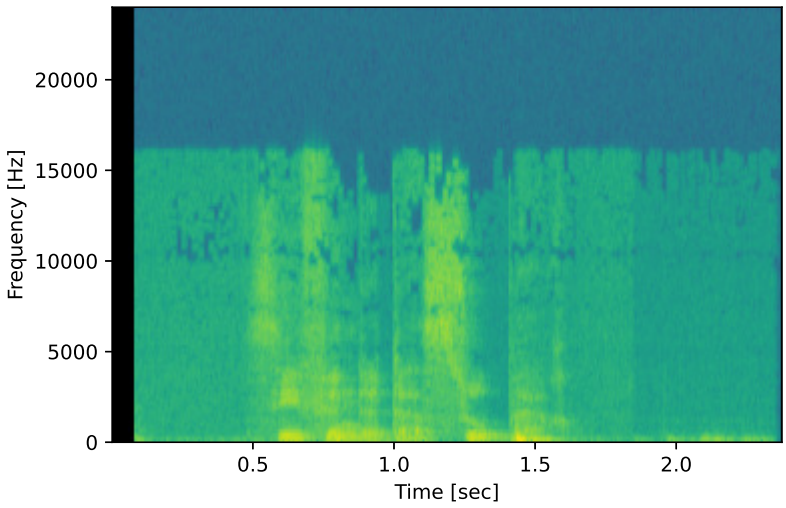

Figura 10 – Espectrograma de um trecho de sinal.

Em seguida, foram criados os arquivos contendo as informações dos bancos de dados, representados na Figura 12 para o exemplo do banco de dados do idioma Alemão.

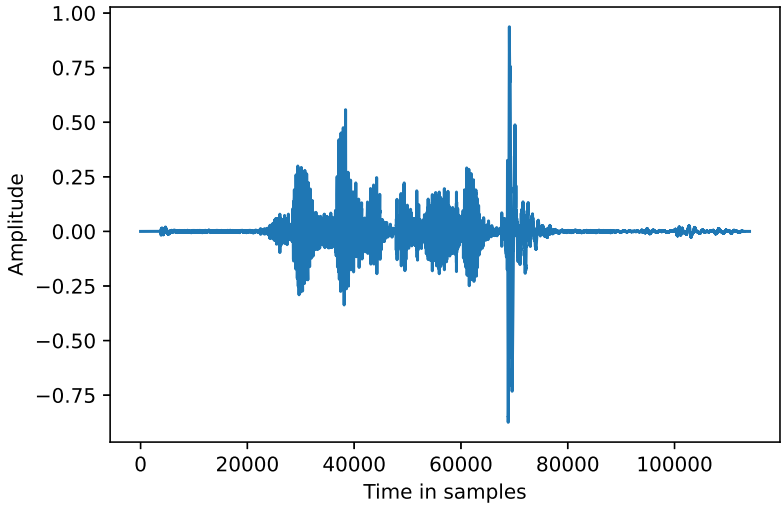

Figura 11 – Visualização de um trecho de sinal no domínio do tempo.

Como o arquivo de áudio foi mantido em um diretório separado, para referenciá-lo, basta selecionar o arquivo pelo nome no diretório de dados.

```
metadata_train_de.info() #informacoes gerais do banco de dados
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 246525 entries, 0 to 246524
Data columns (total 10 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   client_id   246525 non-null  object
 1   path        246525 non-null  object
 2   sentence    246525 non-null  object
 3   up_votes    246525 non-null  int64
 4   down_votes  246525 non-null  int64
 5   age         202696 non-null  object
 6   gender      201479 non-null  object
 7   accent      186431 non-null  object
 8   locale      246525 non-null  object
 9   segment     0 non-null       float64
dtypes: float64(1), int64(2), object(7)
memory usage: 18.8+ MB
```

Figura 12 – Visualização das informações do banco de dados do idioma Alemão

Onde as colunas representam, respectivamente:

i. *client-id*: Chave criptografada do usuário que contribuiu com o áudio;

ii. *path*: Nome do arquivo;

iii. *sentence*: Frase que é dita no áudio;

iv. *up-votes*: Número de votos positivos em relação a qualidade e à clareza do áudio;

v. *down-votes*: Número de votos negativos em relação a qualidade e à clareza do áudio;

vi. *age*: Grupamento de idade do usuário (*teens*, *twenties*, *thirties*, *fourties*, etc.);

vii. *gender*: Gênero do usuário;

viii. *accent*: Sotaque do usuário;

ix. *locale*: Idioma do usuário;

x. *segment*: Classificação do áudio, coluna que foi intencionalmente mantida sem dados e será irrelevante para o estudo;

Para a seleção inicial dos dados foram separadas apenas as colunas ii. e ix., representando o nome do arquivo e o idioma. Para a quantidade de dados foram selecionados apenas 30.000 dados (10.000 para cada idioma) com base em seus valores de *up-votes*, ou seja, seleciona-se os áudios com melhores notas de clareza e qualidade. Essa escolha foi tomada com intuito de priorizar os dados com maior qualidade de clareza e informação, assim que melhor contribuam para a criação de um modelo de classificação assertivo.

Destes 30.000 dados, 13.440 serão aleatoriamente selecionados para o modelo, posteriormente separados em 12.096 para treinamento e 1344 para validação. Enquanto de outro arquivo de cabeçalho destinado apenas para testes foram selecionados os 9.000 (3.000 por idioma) melhores, novamente de acordo com valores de *up-votes* e, desse total, 4.500 são separados aleatoriamente para atuarem como base de dados para teste do modelo gerado.

Em seguida são anexadas as faixas de áudio aos vetores de treinamento, teste e validação, com sua respectiva taxa de amostragem (*samplerate* de 48.000 Hz). Ressalta-se que, para manter a uniformidade dos dados, foram selecionados apenas os três segundos iniciais com presença de algum valor de áudio, desconsiderando ruídos a partir de algum valor limiar, para cada áudio, fazendo com que todos os áudios tenham exatamente três segundos. A codificação desse processo de anexar e cortar os áudios está representado na Figura 13.

Com os vetores de treinamento, validação e teste prontos é preciso calcular os bancos de filtros e MFCC para servirem de entrada ao modelo de rede neural; sendo assim, cria-se um *loop* para os vetores e, com áuxilio da biblioteca *python-speech-features* (Lyons (2013)), com parâmetros corretamente configurados, consegue-se realizar os cálculos. Ao final da execução temos novos vetores de MFCC criados, conforme Figura 14. Uma amostra de MFCC com duas dimensões: número de filtros e janelas MFCC, foi selecionada e exibida na Figura 15. Para auxiliar no entendimento do dado foi exibido em coloração de mapa de calor, portanto dados com maior valor numérico, em comparação com os demais, tem a coloração avermelhada, enquanto valores com menor valor aparecem em tons de azul.

```
limiar = 0.001 # limiar que determina entre audio e ruido

series = []
length = []
for filename in X_train['path'].values:
    fp = path+filename  # filename do áudio
    mp3, samplerate = open_audio(fp)

    i = 0
    while((abs(mp3[i]) < limiar)): #confere o começo de fato do áudio
        i=i+1
    mp3 = mp3[i:i+144000] #seleciona apenas 3s

    while(mp3.shape != (144000,)):
        mp3 = np.append(mp3,0) #caso o audio tenha menos de 3s, completa com zeros

    series.append(mp3)
    length.append(samplerate)

del(fp)
del(mp3)
```

Figura 13 – Lógica implementada para anexar o árquivo de áudio uniformizado aos vetores.

```
MFCC_array = []
for i in range(0,len(X_train)):
    MFCC = python_speech_features.
                        base.logfbank(X_train['Series'][i], samplerate=48000,
                                      winlen=0.025, winstep=0.01, nfilt=20, nfft=2048,
                                                  lowfreq=0, highfreq=cte_highfreq, preemph=0.97)
    MFCC_sc = sc.fit_transform(MFCC)
    MFCC_array.append(MFCC_sc)

del(MFCC)
del(MFCC_sc)

MFCC_array = np.array(MFCC_array)
```

Figura 14 – Lógica implementada para calcular os bancos de filtros e MFCC.

É importante mencionar que as funções do *python-speech-features* dispõem diversos hiper-parâmetros possíveis de serem alterados e ajustados conforme a necessidade e a adequação aos dados de cada problema. Os parâmetros mais sensíveis e que foram base para a realização de diversos testes de implantação foram o tamanho da janela (winlen), passo de janela (winstep) e número de filtros (nfilt).

Ao fim da etapa referente ao pré-processamento de dados, foi preciso gerar um código referente a cada idioma, isso significa transcrever cada idioma em uma combinação de bits para gerar uma correspondência numérica entre o vetor de dados e o vetor de idiomas, inicialmente em forma literal (*string*); esse processo é também conhecido como *one hot encoding*.

Um exemplo de como é estruturado este código está disposto na Figura 16, onde o primeiro dado é do ídioma Francês e foi transcrito para $\{0, 0, 1\}$; o segundo, Alemão, com $\{1, 0, 0\}$; e, por último, Espanhol, com $\{0, 1, 0\}$. Essa mudança auxilia no cálculo dos resultados de predição do modelo, que vai definir pesos para cada posição do vetor de

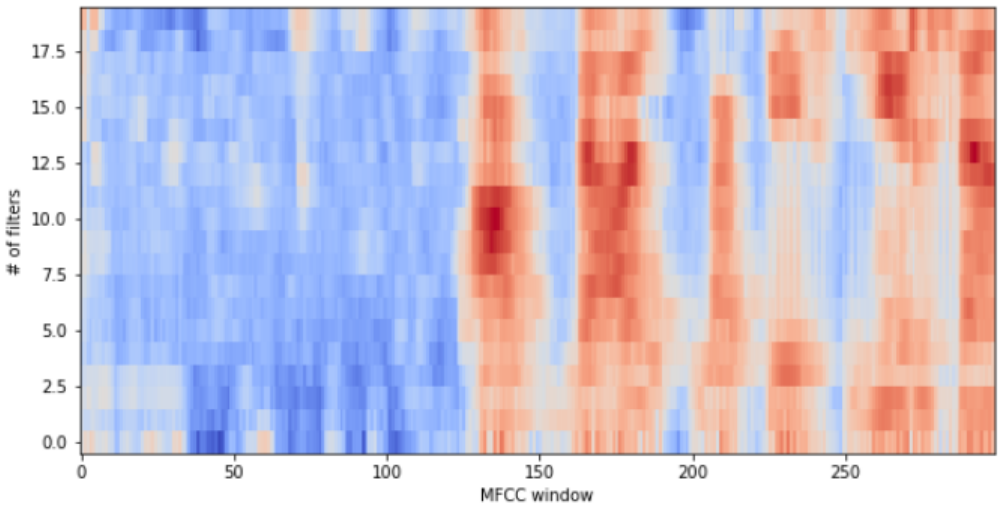

Figura 15 – Representação de uma amostra ampliada do sinal utilizado como entrada na rede neural convolucional.

saída do modelo com o código de dados para determinar qual idioma foi falado.

```
language_dummies[:5]
```

```
array([[0, 0, 1],
       [1, 0, 0],
       [1, 0, 0],
       [0, 1, 0],
       [0, 1, 0]], dtype=uint8)
```

Figura 16 – Exemplo do dicionário gerado para três idiomas diferentes.

3.2 Criação da Rede Neural Convolucional

Conforme mencionado no capítulo 1, uma das motivações e inspirações para essa implementação foi o trabalho (Fernandez, Lucero Guadalupe, 2020), portanto o primeiro escopo de rede neural convolucional foi baseado nesse projeto, que utiliza recursos do *Tensor Flow* e mais especificamente da biblioteca *keras* (TENSORFLOW, 2020) para desenho da rede e configuração de parâmetros básicos de funcionamento.

A codificação da rede criada foi realizada seguindo os parâmetros da Figura 17, onde tem-se 5 camadas de redes convolucionais seguindo o tamanho $(7, 7)$, $(5, 5)$ e $(3, 3)$ de acordo com as camadas para o dado de entrada. Enquanto a configuração dos parâmetros do modelo e de como será feito o treinamento está seguindo a Figura 18, onde são definidos o algoritmo de otimização do modelo (*Adam*), a taxa de aprendizagem inicial (*initial_lrate*), a taxa de dados descartados após cada época (*drop*), a fórmula de taxa de aprendizagem a ser seguida (*lrate*), dados de treinamento e validação, quantidade de épocas (*epochs*) e o tamanho de cada lote de dados (*batch_size*).

```
input_shape = (299,20,1)
model = Sequential()

model.add(Conv2D(32,(7, 7), activation='selu', padding='valid', input_shape=input_shape))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(3,3), strides=2, padding='same'))
model.add(Conv2D(64,(5,5), activation='selu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(3,3), strides=2, padding='same'))
model.add(Conv2D(128,(3,3), activation='selu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(3,3), strides=2, padding='same'))
model.add(Conv2D(256,(3,3), activation='selu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(3,3), strides=2, padding='same'))
model.add(Conv2D(512,(3,3), activation='selu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(3,3), strides=2, padding='same'))
model.add(Flatten())
model.add(BatchNormalization())
model.add(Dense(256, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(3, activation='softmax'))
```

Figura 17 – Lógica implementada para desenho da configuração da rede neural convolucional.

Portanto, correlacionando com os aspectos teóricos de redes convolucionais abordados anteriormente, está sendo realizada a divisão em 5 camadas de *convolution* + *pooling*, aplicando a técnica *padding valid* na primeira camada e *padding same* nas demais camadas, para diminuição do número de *features* e, em seguida, refinamento do mapa de *features*, e processo de classificação composto por duas camadas, uma com ativação *relu*, função de ativação linear, e a última com ativação *Softmax*, que prioriza componentes mais importantes para o vetor de saída conforme as execuções do algoritmo, função de ativação baseada em uma função de distribuição de probabilidades.

Além dos parâmetros disponibilizados nas Figuras 17 e 18, vale ressaltar que diversos outros testes foram realizados alterando todos os parâmetros envolvidos na criação do modelo e na análise dos dados; alguns deles estão nas Tabelas 1 e 2, incluindo diferenças no pré-processamento de dados, variação no cálculo da MFCC, mudanças de parâmetros no desenho da rede neural convolucional, diferenças nos métodos de ativação da rede neural, variação de épocas para o processamento do modelo, etc. Mais testes foram realizados, mas não serão exibidos neste relatório para não sobrecarregar a leitura. Sendo o conjunto de parâmetros exibido na seção de Resultados, o que chegou às melhores métricas combinando o desempenho do modelo e o desempenho computacional.

3.3 Métricas de Desempenho e Resultados

De acordo com a Seção 2.6 as métricas a serem implementadas são as usuais para análise de modelos de redes neurais em Machine Learning. Para a geração da matriz de

```
import math
from keras.callbacks import LearningRateScheduler
adam = Adam()
def step_decay(epoch):
    # 00158 = 90.4%
    initial_lrate = 0.00158
    drop = 0.9
    epochs_drop = 1
    lrate = initial_lrate * math.pow(drop, math.floor((1+epoch)/epochs_drop))
    return lrate


model.compile(loss='categorical_crossentropy',optimizer=adam,metrics=['accuracy'])



checkpoint = ModelCheckpoint(
                'model.h5',
                monitor='val_acc',
                verbose=0,
                save_best_only=True,
                mode='max'
                )

lrate = LearningRateScheduler(step_decay)
#es = EarlyStopping(monitor='val_loss',mode = 'max')
model.fit(
                X_train_MFCC,
                y_train_MFCC,
                epochs=40,
                callbacks=[checkpoint, lrate],
                verbose=1,
                validation_data=(X_validation_MFCC, y_validation_MFCC),
                batch_size=32)
```

Figura 18 – Lógica implementada para determinar os parâmetros de configuração do modelo, treinamento e validação.

confusão é utilizado a própria função da Biblioteca Sklearn (SCIKIT-LEARN, 2007-2020), demonstrada com um exemplo na Figura 19.

		Idioma Predito		
		Alemão	Espanhol	Francês
Idioma real	Alemão	80.5	7.8	11.7
	Espanhol	8.7	80.0	11.3
	Francês	10.9	9.6	79.5

Figura 19 – Matriz de Confusão de exemplo, em percentual, para ilustração das métricas de resultado do sistema.

Os dados referentes a cálculo de Acurácia, Precisão, Recall e F-Score também são calculados com áuxilio da Biblioteca Sklearn, utilizando uma função chama de *classification-report*, demonstrada na Figura 20, é possível visualizar o resultado das diferentes métricas

para cada idioma, junto com a acurácia geral do modelo as médias de acurácia para cada caso.

```
print(classification_report(y_pred_test,y_test))
```

	precision	recall	f1-score
0	0.80	0.80	0.80
1	0.83	0.80	0.81
2	0.77	0.80	0.78
accuracy			0.80
macro avg	0.80	0.80	0.80
weighted avg	0.80	0.80	0.80

Figura 20 – Precisão, Revocação, *F1-Score* e Acurácia gerado para o modelo.

Tabela 1 – Parte 1: Descrição dos testes de diferentes parâmetros realizados no trabalho.

Descrição do modelo de teste	Dados de treinamento/ validação/teste	Acurácia de validação	Acurácia de testes
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), separação de 10% para validação e sem alterações no modelo da rede	12096/1344/4500	0,8936	0,78
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), separação de 30% para validação e sem alterações no modelo da rede	9408/4032/4500	0,8790	0,77
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), separação de 50% para validação e sem alterações no modelo da rede	6720/6720/4500	0,8609	0,75
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), número de filtros aumentado para 60, separação de 10% para validação e sem alterações no modelo da rede	12096/1344/4500	0,8780	0,78
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), número de filtros aumentado para 60, separação de 30% para validação e sem alterações no modelo da rede	9408/4032/4500	0,8800	0,77
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), número de filtros diminuído para 20, separação de 10% para validação e sem alterações no modelo da rede	12096/1344/4500	0,9033	0,78
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), número de filtros diminuído para 20, separação de 30% para validação e sem alterações no modelo da rede	9408/4032/4500	0,8802	0,78
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), alterado para cálculo linear de MFCC (diferente de logarítmica), separação de 30% para validação e sem alterações no modelo da rede	9408/4032/4500	0,8445	0,72

Tabela 2 – Parte 2: Continuação da descrição dos testes de diferentes parâmetros realizados no trabalho.

Descrição do modelo de teste	Dados de treinamento/ validação/teste	Acurácia de validação	Acurácia de testes
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), alterado para cálculo linear de MFCC (diferente de logarítmica), separação de 50% para validação e sem alterações no modelo da rede	6720/6720/4500	0,8141	0,69
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), alterado para cálculo linear de MFCC (diferente de logarítmica), separação de 10% para validação e sem alterações no modelo da rede	12096/1344/4500	0,8579	0,74
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), separação de 30% para validação e com alteração do parâmetro de ativação da classificação da rede para 'selu' invés de 'relu'	9408/4032/4500	0,8755	0,77
Dados de entrada iguais para cada idioma, vetores de áudio com 3s e preenchidos com zero na ausência de dado (s/tratamento), separação de 10% para validação e com alteração do parâmetro de ativação da classificação da rede para 'selu' invés de 'relu'	12096/1344/4500	0,8943	0,79
Dados de entrada iguais para cada idioma, vetores de áudio com 3s a partir de um limiar, separação de 10% para validação e com sem alteração de parâmetros na rede	12096/1344/4500	0,8927	0,80

4 Resultados preliminares e Discussão

Considerando o modelo gerado e discutido nas seções anteriores, o seu desempenho será analisado baseando-se no resultado referente às métricas da seção 2.6 para diferentes casos: utilizando a base de dados separada para validação, utilizando a base de dados separada para testes e, por fim, em testes livres com áudios retirados da Internet.

4.1 Resultados de Validação

Para os resultados gerados a partir da validação é esperado que a acurácia encontrada seja a maior entre os resultados, isso é devido ao banco de dados de validação ser retirado como uma parcela do banco de dados de treinamento e, ainda assim, a parte de validação ser utilizada para validar o modelo durante o treinamento, configurados no Keras (TENSORFLOW, 2020).

A matriz de confusão gerada na etapa de validação está exibida em valores percentuais na Figura 21; nela é possível perceber que o número de acertos de idioma, ou seja, a acurácia, está por volta de 89%, ultrapassando 90% de acertos para o primeiro idioma, o alemão. Quanto aos erros de predição no sistema, tem-se que variou entre 3% e 7.5% uniformemente para os dados, o que pode representar que a taxa de erro ocorra em dados difíceis de predizer, seja por possuírem características de mais de um idioma, serem sinais não vozeados, etc.

		Idioma Predito		
		Alemão	Espanhol	Francês
Idioma real	Alemão	90.6	3.1	6.2
	Espanhol	3.8	88.8	7.4
	Francês	5.6	6.0	88.4

Figura 21 – Matriz de confusão, em valores percentuais, para a base de dados de validação.

Também é possível gerar os valores das outras métricas utilizando a função *classification-report*. Para os resultados com a base de dados de validação, demonstrados na Figura 22, é possível perceber que alguns valores chegaram a ultrapassar a marca de 90%, o que são ótimos resultados, ainda assim, a média de acurácia se manteve em 0.89, junto com as médias ponderadas das outras métricas. Novamente percebe-se que a

variação entre as métricas de precisão, revocação e *f1-score*, para os dados de validação, estão uniformemente distribuidas entre os dados.

```
print(classification_report(y_val,y_pred_val))
```

	precision	recall	f1-score
0	0.91	0.91	0.91
1	0.91	0.89	0.90
2	0.87	0.88	0.88
accuracy			0.89
macro avg	0.89	0.89	0.89
weighted avg	0.89	0.89	0.89

Figura 22 – Precisão, Revocação, *F1-Score* e Acurácia para a base de dados de validação.

4.2 Resultados de Teste

Para os resultados com a base de dados de testes, conforme mencionado anteriormente, foram inseridos logo após a geração do modelo e resultados com a base de dados de validação. Os dados foram preditos com aúxilio da função *predict* do *Keras*, e o vetor de predições gerado foi comparado ao dicionário de dados, código *one hot encoding*, para definir seu desempenho. Os dados referentes à matriz de confusão, em percentual, estão disponíveis na Figura 23. Pode-se perceber que os valores de acertos diminuíram em relação aos resultados com os dados de validação, mas a acurácia média ainda é alta–cerca de 80%.

No geral, não houve mudanças significativas produzidas por diferentes quantidades de dados preditos para os diferentes idiomas abordados no produto, assim como na validação; portanto, não há uma evidência clara de qual seja o ponto de melhoria nesse caso. Uma abordagem já realizada foi a tentativa de alterar diversos parâmetros de pré-processamento, número de bancos de filtros e de parâmetros da rede convolucional; porém, os melhores resultados ainda apontam para essa média de acurácia.

		Idioma Predito		
		Alemão	Espanhol	Francês
Idioma real	Alemão	80.5	7.8	11.7
	Espanhol	8.7	80.0	11.3
	Francês	10.9	9.6	79.5

Figura 23 – Matriz de confusão, em valores percentuais, para a base de dados de teste.

Para o resultado das demais métricas; precisão, revocação e *f1-score*; percebe-se resultados parecidos com os obtidos na matriz de confusão, disponíveis na Figura 24, valores, no geral, entre 77% e 81% e com a acurácia média do sistema em 80%, representando resultados satisfatórios para o modelo gerado, com capacidade de serem melhorados aumentando a base de treinamento ou a complexidade da rede neural modelada, por exemplo, mas exigindo poder computacional maior.

```
print(classification_report(y_pred_test,y_test))
```

	precision	recall	f1-score
0	0.80	0.80	0.80
1	0.83	0.80	0.81
2	0.77	0.80	0.78
accuracy			0.80
macro avg	0.80	0.80	0.80
weighted avg	0.80	0.80	0.80

Figura 24 – Precisão, Revocação, *F1-Score* e Acurácia para a base de dados de teste.

4.3 Resultados Adicionais

Além de analisar o comportamento do modelo de rede convolucional gerado para as bases de dados de validação e teste, decide-se ampliar a análise do modelo gerado para predizer dados reais retirados de vídeos do *Youtube* e testar a acurácia do modelo para esses casos, sendo uma possibilidade de simular o comportamento do modelo em uma situação cotidiana.

Para isso, foram selecionados trechos de áudios contendo frases dos diversos idiomas contidos neste trabalho, eliminou-se ruídos e aplicou-se os parâmetros de pré-processamento realizados neste trabalho, assim o áudio processado é lido pelo modelo gerado e a resposta da predição é um vetor com n pesos, onde n é o número de idiomas do sistema, e a posição do vetor que mais se aproximar de 1 é o idioma predito de acordo com o dicionário dos dados.

Foram selecionados áudios provindos de três videos do *Youtube*, um para cada idioma, sendo que o foco da pesquisa envolveu reunir áudios com frases claras de cada idioma, dita por autores dos gêneros feminino e masculino e que sejam curtas, para se assemelhar ao máximo aos casos da base de dados *Common Voice* utilizada na criação do modelo. Foram selecionados áudios de vídeos que lecionavam cada idioma com duração de aproximadamente 10 segundos inicialmente, cerca de 15 áudios/idioma, totalizando 45 áudios:

- Espanhol: Para o idioma Espanhol foram retirados trechos do vídeo "*[Conversation at a Restaurant] Spanish (from Spain) Speaking Example*", disponível no *Youtube* (TALK, 2021), que mostra um diálogo fictício entre três atores: garçom e casal de clientes. Nesse vídeo é simulado um diálogo que envolve a chegada dos clientes ao restaurante, pedido de pratos e bebidas do cardápio, até a despedida do casal do restaurante (Figura 25).

Figura 25 – Video utilizado como fonte de Espanhol para o teste adicional. Fonte: Reprodução Youtube (TALK, 2021)

- Francês: Para o idioma Francês foram retirados trechos do vídeo "*At the hotel French conversation*", disponível no *Youtube* (DEMéO, 2021), que mostra um diálogo fictício entre um recepcionista de hotel e uma hóspede recém-chegada. São realizadas as introduções de cada personagem e depois acordada a alocação de um dos quartos do hotel (Figura 26).

- Alemão: Para o idioma Alemão foram retirados trechos do vídeo "*Learn German with videos*", disponível no *Youtube* (LINGUATV.COM, 2021), que mostra um diálogo fictício entre um homem e uma mulher que eventualmente se esbarraram na rua, abordando pontos de introdução dos personagens e temas básicos, como, por exemplo, passar o número de telefone ou dizer onde reside na cidade (Figura 27).

Após separados os respectivos áudios de cada vídeo, a etapa de pré-processamento seguiu a ordem de execução da rotina principal do produtor: foram criados arquivos de cabeçalho semelhantes ao da Figura 12, separados entre dados de treino e teste, mesmo que apenas os dados de teste fossem usados, para garantir aleatoriedade, e, por último, anexados os arquivos de áudio unificados em 3 segundos a partir do mesmo limiar. Considerando que

Figura 26 – Video utilizado como fonte de Francês para o teste adicional. Fonte: Reprodução Youtube (DEMéO, 2021)

Figura 27 – Video utilizado como fonte de Alemão para o teste adicional. Fonte: Reprodução Youtube (LINGUATV.COM, 2021)

a base de dados completa conta com cerca de 45 áudios, de fato, apenas a metade participa de cada ciclo de execução de teste; assim temos resultados heterogêneos e podemos analisar melhor a rede neural gerada.

Para a etapa de teste foram carregados os dados gerados no pré-processamento, calculados os respectivos bancos de filtros e MFCC, e também criado o dicionário de dados. Por fim, os dados entraram no modelo de rede neural convolucional criado para o projeto

principal e seus resultados preditos foram comparados aos idiomas reais.

Os resultados gerados para a base de dados livre estão dispostos na Figura 28, onde a primeira linha representa o idioma alemão, a segunda o idioma espanhol e a terceira o idioma francês. Recapitulando, a interpretação correta da matriz de confusão é a seguinte:

- a_{ii} - Elemento predito corretamente como sendo do idioma i considerando que seu idioma é de fato i;
- a_{ij} - Elemento predito incorretamente como sendo do idioma j considerando que seu idioma na verdade é i.

		Idioma Predito		
		Alemão	Espanhol	Francês
Idioma real	Alemão	100.0	0.0	0.0
	Espanhol	11.1	77.8	11.1
	Francês	30.0	0.0	70.0

Figura 28 – Matriz de confusão, em valores percentuais, para a base de dados de teste livre.

Assim sendo, observa-se que os resultados gerados acertaram todos os casos onde classificou o idioma como alemão, conforme é possível observar na primeira linha da matriz da Figura 28, enquanto apresentou alguns erros para a classificação do idioma espanhol ou francês, linhas 2 e 3, respectivamente.

Outro ponto interessante a se notar é a primeira coluna, que representa a coluna onde o áudio era alemão. Se isolarmos a análise da matriz gerada é possível perceber que o sistema cometeu a maioria dos erros apenas envolvendo áudios cujo idioma era alemão, ou seja, o elemento a_{21} foi predito como sendo espanhol mas era alemão e o elemento a_{31} foi predito como francês mas era alemão.

Por fim, os resultados com as principais métricas de análise estão disponíveis na Figura 29, onde é possível perceber que a principal métrica considerada, que é a acurácia, se manteve na média do projeto, mesmo tendo erros relacionados ao idioma alemão.

Além disso, é possível dizer que os resultados gerados corroboram o fato do modelo criado ser plástico a ponto de manter as métricas de desempenho mesmo alterando-se os dados de entrada, além de que abre possibilidade para serem realizados diversos outros testes, como a expansão dessa base de dados livre, a inclusão de um idioma não abordado

```
print(classification_report(y_pred_ft,y_ft))
```

	precision	recall	f1-score
0	0.43	1.00	0.60
1	1.00	0.78	0.88
2	0.88	0.70	0.78
accuracy			0.77
macro avg	0.77	0.83	0.75
weighted avg	0.87	0.77	0.79

Figura 29 – Precisão, Revocação, *F1-Score* e Acurácia para a base de dados de teste livre.

até o momento para observar o comportamento do sistema ou até a inclusão de áudios não necessariamente associados a sinais de fala nítida, como canções ou poesias.

5 Conclusões

Os resultados atingidos cumprem com a proposta de projeto de criar um sistema que identifique o idioma a partir de um sinal curto de áudio e, principalmente, com a acurácia obtida, tempo e complexidade computacional, é capaz de ser implementado em um sistema real. Para implementação em um projeto real uma possibilidade de melhoria seria gerar rotinas de treinamento e aperfeiçoamento baseado em casos reais, principalmente em casos de erros do sistema, além de aumentar a robustez da rede a partir de melhorias pontuais no código e na base de dados, como, por exemplo, usar estruturas de dados mais eficientes para diminuir a manipulação de vetores, assim, diminuindo o tempo computacional do modelo e, consequentemente, aumentando a sua capacidade de análise de dados.

Todo o processo criativo de realização deste projeto foi realizado durante o ano de 2021 e seguindo o cronograma de execução (apêndice A), iniciando na proposta inicial e elaboração do projeto; busca por bibliografias e ferramentas técnicas; implementação prática de pré-processamento, modelo de rede neural convolucional e métricas de desempenho e a parte de testes adicionais e melhorias; dividido durante o ano letivo com as respectivas datas de início e término para cada item.

Possíveis próximos passos para aperfeiçoar o sistema seriam a adição de mais idiomas, melhorar a estrutura e a capacidade da rede neural para se tornar mais precisa e com mais eficácia, principalmente em idiomas que possuem pouca diferenciação na entonação de fala, ou até diferenciação de sotaques para a mesma língua conforme demonstrado em (KUMAR et al., 2004), e consequentemente possuem formas de onda parecidas. Além de implementar estruturas de dados mais eficientes e que permitam manipulações de dados de forma mais eficiente. Por último, pode-se adicionar o modelo gerado neste projeto em um sistema ou aplicativo já existente, se tornando parte integrante de um sistema maior onde as predições sejam bases para outras ações do sistema, como tradução simultânea ou análise de outros parâmetros da fala.

Para um futuro projeto de pós-graduação também é possível estudar o comportamento do sistema com múltiplas entradas gerando respostas em tempo real para um núcleo de processamento, por exemplo, simular o processamento de diversos microfones espalhados em uma rua para um evento de muita movimentação popular como o Carnaval de rua em São Paulo. Nesse caso além da adaptação do sistema para múltiplas entradas seria preciso tratar a potência recebida de cada sinal separadamente, aumentando consideravelmente a capacidade de processamento e análise dos dados em tempo real, além de definir melhores parâmetros e condições de análise para cada parâmetro do modelo, assim tornando-se mais assertivo em um cenário totalmente diferente do estudado neste trabalho.

Referências

AGRAWAL, P.; KAUR, H.; KAUR, G. Multi lingual speaker identification on foreign languages using artificial neural network. v. 5713, p. 975–8887, 11 2012. Citado na página 3.

ALIM, S. A.; RASHID, N. K. A. Some commonly used speech feature extraction algorithms. In: LOPEZ-RUIZ, R. (Ed.). *From Natural to Artificial Intelligence*. Rijeka: IntechOpen, 2018. cap. 1. Disponível em: <https://doi.org/10.5772/intechopen.80419>. Citado 2 vezes nas páginas 6 e 9.

Amazon. *Alexa Enterprise and Business Solutions*. 2021. [Online; accessed 6-August-2021]. Disponível em: <https://developer.amazon.com/en-US/alexa/enterprise-and-business>. Citado na página 2.

Amazon. *Amazon Transcribe*. 2021. [Online; accessed 6-August-2021]. Disponível em: <https://aws.amazon.com/pt/transcribe/>. Citado na página 2.

Amazon, AWS Official Documentation. *Identifying the dominant language in your media files*. 2021. [Online; accessed 04-November-2021]. Disponível em: <https://docs.aws.amazon.com/transcribe/latest/dg/auto-lang-id.html>. Citado 2 vezes nas páginas 6 e 3.

Baheti, Pragati. *A Comprehensive Guide to Convolutional Neural Networks*. 2021. [Online; accessed 04-November-2021]. Disponível em: <https://www.v7labs.com/blog/convolutional-neural-networks-guide>. Citado 3 vezes nas páginas 6, 12 e 13.

CHAKROBORTY, S.; ROY, A.; SAHA, G. Fusion of a complementary feature set with mfcc for improved closed set text-independent speaker identification. In: *2006 IEEE International Conference on Industrial Technology*. [S.l.: s.n.], 2006. p. 387–390. Citado na página 8.

CHOUGULE, S.; REGE, P. Language independent speaker identification. In: . [S.l.: s.n.], 2007. p. 364–368. Citado 2 vezes nas páginas 3 e 4.

CRIPTOGRAPHY, P. *Mel Frequency Cepstral Coefficient (MFCC) tutorial*. 2013. Practical Criptography. Disponível em: <http://www.practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/>. Acesso em: 04 apr. 2021. Citado na página 8.

DEMéO, R. *À l'hotel: conversation en français / At the hotel: French conversation*. 2021. Disponível em: <https://youtu.be/geyHn8Ai6NM>. Acesso em: 20 sep. 2021. Citado 3 vezes nas páginas 7, 28 e 29.

Eric W., Weisstein. *Convolution, from MathWorld*. 2021. [Online; accessed 02-November-2021]. Disponível em: <https://mathworld.wolfram.com/Convolution.html>. Citado na página 11.

Fernandez, Lucero Guadalupe. *Spoken language Detection*. 2020. [Online; accessed 17-July-2021]. Disponível em: <https://www.kaggle.com/lucerofernandez/spoken-language-detection>. Citado 2 vezes nas páginas 4 e 19.

GELLY, G.; GAUVAIN, J. Spoken language identification using lstm-based angular proximity. 08 2017. Citado na página 4.

HAYKIN, S. *Neural Networks and Learning Machines*. [S.l.]: Pearson, 2009. Citado 3 vezes nas páginas 6, 9 e 10.

IBM. *Watson Speech to Text (STT)*. 2021. [Online; accessed 6-August-2021]. Disponível em: <https://www.ibm.com/br-pt/cloud/watson-speech-to-text>. Citado na página 2.

Julien Simon, AWS News Blog. *Amazon Transcribe Now Supports Automatic Language Identification*. 2021. [Online; accessed 6-August-2021]. Disponível em: <https://aws.amazon.com/pt/blogs/aws/amazon-transcribe-now-supports-automatic-language-identification/>. Citado na página 3.

KUMAR, C. et al. Language identification for multilingual speech recognition systems. 01 2004. Citado 2 vezes nas páginas 4 e 32.

Lauren Barack, GearBrain. *How to train Google Assistant to learn your voice through the Google Home app*. 2019. [Online; accessed 04-November-2021]. Disponível em: <https://www.gearbrain.com/train-google-assistant-learn-voice-2638801924.html>. Citado 2 vezes nas páginas 6 e 1.

LAZZARI, G. *Speaker-Language Identification and Speech Translation*. 1999. Carnegia Mellon University website. Disponível em: <https://www.cs.cmu.edu/~ref/mlim/chapter7.html>. Acesso em: 04 apr. 2021. Citado na página 1.

LI, B. M. H.; LEE, C.-H. A vector space modeling approach to spoken language identification. *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, v. 15, n. 1, p. 271–284, 2007. Citado na página 4.

LI, F. F.; COX, T. J. *Digital Signal Processing in Audio and Acoustical Engineering*. [S.l.]: CRC Press, 2019. Citado 3 vezes nas páginas 6, 8 e 10.

LINGUATV.COM. *Deutsch lernen mit Videos / Learn German with videos!* 2021. Disponível em: <https://youtu.be/nd0Y_iIaJns>. Acesso em: 20 sep. 2021. Citado 3 vezes nas páginas 7, 28 e 29.

LYONS, J. *'python_speech_features' Official Documentation*. 2013. Python_speech_features. Disponível em: <https://python-speech-features.readthedocs.io/en/latest/>. Acesso em: 04 apr. 2021. Citado 2 vezes nas páginas 9 e 17.

MCLOUGHLIN, I. *Applied Speech and Audio Processing*. [S.l.]: Cambridge, 2009. Citado 2 vezes nas páginas 6 e 7.

MOZILLA. *Common Voice*. 2021. Mozilla Official Website. Disponível em: <https://commonvoice.mozilla.org/en/about>. Acesso em: 04 apr. 2021. Citado 2 vezes nas páginas 3 e 15.

Saha, Sumit. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way*. 2018. [Online; accessed 04-November-2021]. Disponível em: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>. Citado na página 11.

SCIKIT-LEARN. *'sklearn.metrics.confusion_matrix' Official documentation*. 2007–2020. Scikit-learn Official Documentation. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html>. Acesso em: 04 apr. 2021. Citado 2 vezes nas páginas 13 e 21.

SHIN, H.; CHO, J. Unconstrained snoring detection using a smartphone during ordinary sleep. *Biomedical engineering online*, v. 13, p. 116, 08 2014. Citado na página 6.

TALK, P. . *[Conversation at a Restaurant] Spanish (from Spain) Speaking Example*. 2021. Disponível em: <https://youtu.be/GI28zqFOSVk>. Acesso em: 20 sep. 2021. Citado 2 vezes nas páginas 7 e 28.

TENSORFLOW. *'Keras' Official documentation*. 2020. Keras Official Documentation. Disponível em: <https://www.tensorflow.org/guide/keras?hl=pt-br>. Acesso em: 04 apr. 2021. Citado 2 vezes nas páginas 19 e 25.

Timekettle. *WT2 Plus AI Realtime Translator Earbuds*. 2021. [Online; accessed 06-August-2021]. Disponível em: <https://www.timekettle.co/products/wt2-plus>. Citado na página 2.

Wikipedia contributors. *MP3 — Wikipedia, The Free Encyclopedia*. 2021. [Online; accessed 7-April-2021]. Disponível em: <https://en.wikipedia.org/w/index.php?title=MP3&oldid=1014824003>. Citado na página 5.

APÊNDICE A – Cronograma de Execução do Trabalho de Graduação

O Trabalho de Graduação estruturou-se nos seguintes tópicos a seguir, que foram essenciais para dividir as tarefas de criação do presente trabalho e alocadas durante o ano de elaboração:

1. **Proposta inicial, ajustes e elaboração do plano de projeto**: Primeira ideia de implementação do projeto, que foi discutida, refinada e ajustada com aúxilio do orientador para cumprir a proposta pedagógica do Trabalho de Graduação;
2. **Busca por bibliografias, livros e implementações similares**: A partir do escopo inicial definido foi preciso buscar referências que servissem de apoio e estudo para colaborar na implementação do sistema e construir bases teóricas sólidas para resolução do problema;
3. **Busca por ferramentas técnicas e bancos de dados**: Além da compreensão da teoria envolvida foi preciso conhecer as bibliotecas de pré-processamento, análise de áudios e criação de redes neurais convolucionais, além de conhecer também bancos de dados livres com áudios disponíveis para *download*, com suas características e organizações próprias de dados;
4. **Implementação do pré-processamento dos dados**: O primeiro passo da implementação prática foi compreender os detalhes do banco de dados e adotar um padrão de áudio, com tempo definido e limiar de inicio para cada áudio, eliminando qualquer ruído. Após esse processo foi preciso realizar a divisão entre treinamento, validação e teste, anexar os árquivos de áudio a cada linha do vetor e calcular bancos de filtros e MFCC.
5. **Implementação do modelo de rede neural convolucional**: Assim que os dados estão organizados é preciso criar, treinar, validar e testar o modelo gerado, com os coeficientes otimizados para gerar melhores métricas de desempenho;
6. **Implementação de métricas de análise de resultados**: Para testar o modelo e gerar dados de saída, foi preciso definir e implementar métricas de análise e condições para o modelo cumprir de modo a ter parâmetros de referência para aumentar a performance do modelo;
7. **Melhorias no pré-processamento e rede neural**: Após a criação do sistema e dos primeiros resultados do modelo, foi preciso fazer ajustes finos no modelo para

aumentar o desempenho das métricas estabelecidas, além de variar parâmetros de configuração e testar novas metodologias de pré-processamento dos dados, criação do modelo e criação de métricas de desempenho. Essa etapa visou realizar modificações na forma com que os áudios eram selecionados, nas características da função MFCC, nas características da rede neural e também na forma com que analisariamos os resultados;

8. **Implementação de testes adicionais com bases de dados livres**: Adicionalmente ao teste do modelo foi implementado um teste livre, com áudios retirados de videos da Internet, para conferir a capacidade de generalização da rede em ambiente real e qual seria o comportamento com áudios "fora do padrão" do banco de dados inicialmente utilizado;

9. **Escrita técnica das etapas de construção do Trabalho de Gradução**: Elaboração da documentação referente a toda pesquisa, implementação e aspectos técnicos que envolveram a criação do sistema;

10. **Elaboração de apresentação do projeto para a Banca Avaliativa**: Além da escrita técnica é necessário criar apresentação que sumarize toda a pesquisa feita de uma forma sequencial e didática para introduzir o trabalho a alguem que não tenha nenhum parâmetro de referência sobre a área ou a metodologia utilizada.

O andamento do trabalho no ano de 2021 seguiu conforme a Tabela 3 abaixo (os números à esquerda indicam as tarefas enumeradas na presente seção):

Tabela 3 – Cronograma de acompanhamento do Trabalho de Graduação.

	jan	fev	mar	abr	mai	jun	jul	ago	set	out	nov
1	x										
2		x	x								
3		x	x								
4			x	x							
5				x	x						
6					x						
7						x	x	x			
8							x	x	x	x	
9				x	x	x	x	x	x	x	x
10											x