



SEPARAÇÃO DE FONTES SONORAS AUXILIADA POR DEEP LEARNING

Jéssica Richards Nascimento

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Carlos Eduardo Pedreira

Rio de Janeiro
Dezembro de 2021

SEPARAÇÃO DE FONTES SONORAS AUXILIADA POR DEEP LEARNING

Jéssica Richards Nascimento

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientador: Carlos Eduardo Pedreira

Aprovada por: Prof. Carlos Eduardo Pedreira

Prof^ª. Carolina Gil Marcelino

Prof. Hugo Tremonte de Carvalho

RIO DE JANEIRO, RJ – BRASIL

DEZEMBRO DE 2021

Richards Nascimento, Jéssica

Separação de fontes sonoras auxiliada por Deep Learning/Jéssica Richards Nascimento. – Rio de Janeiro: UFRJ/COPPE, 2021.

XI, 42 p.: il.; 29,7cm.

Orientador: Carlos Eduardo Pedreira

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2021.

Referências Bibliográficas: p. 40 – 42.

1. separação de fontes sonoras. 2. monoaural.
3. deep-learning. I. Pedreira, Carlos Eduardo.
- II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

*Dedico esta dissertação a todos
que estiveram comigo nesse
caminho*

Agradecimentos

Agradeço a minha família por ter me acompanhado durante todos esses anos e ter possibilitado essa jornada.

Agradeço a todos os professores que me ensinaram, sem cada um de vocês não conseguiria escrever esse trabalho. Um agradecimento especial ao professor Pedreira que aceitou orientar esse trabalho.

Obrigada aos professores Carolina Marcelino e Hugo Carvalho por terem aceitado participar da banca.

Agradeço a CAPES pela bolsa de mestrado, possibilitando essa dissertação.

Agradeço a todos os meus colegas de laboratório que me ajudaram tanto com sugestões e material para o trabalho, quanto em ideias de qual caminho seguir com a tese.

Agradeço também aos meus amigos por me apoiarem nessa jornada.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

SEPARAÇÃO DE FONTES SONORAS AUXILIADA POR DEEP LEARNING

Jéssica Richards Nascimento

Dezembro/2021

Orientador: Carlos Eduardo Pedreira

Programa: Engenharia de Sistemas e Computação

Um stream auditivo é um grupo de sons que no entendimento humano pertencem à mesma cena. O uso de máscaras binárias para separar uma cena auditiva em dois ou mais streams tem se mostrado muito efetivo. Abordagens mais recentes usam métodos de aprendizado supervisionado para gerar essas máscaras. Os áudios utilizados nos experimentos foram gerados artificialmente, uma mistura de vogal falada e outro áudio. O trabalho utiliza esses áudios monoaurais, propondo encontrar uma máscara binária para o stream de interesse. Para encontrar essas máscaras duas abordagens foram utilizadas: a primeira trabalha com os coeficientes de frequência mel e rede neural convolucional, e a segunda com os espectrogramas dos áudios e uma rede U-Net. A primeira abordagem não se mostrou muito efetiva. A segunda apresentou melhores resultados.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

SOUND SOURCE SEPARATION AUXILIATED DEEP LEARNING MODELS

Jéssica Richards Nascimento

December/2021

Advisor: Carlos Eduardo Pedreira

Department: Systems Engineering and Computer Science

An auditory stream is a group of sounds that in human perception belong to the same scene. The use of binary masks to segregate an auditory scene in two or more streams has shown to be very effective. More recent approaches use supervised learning models to create these binary masks. The audios used in the experiments were artificially created, a mixture of vowel sound and other audio. The work uses these monaurals audios, proposing to find a binary mask for the stream of interest. To find these masks two approaches were explored: the first one uses the mel frequency cepstral coefficients and the convolutional neural network, and the second one uses the audios spectrograms and a U-Net network. The first approach wasn't very effective. The second presented better results.

Sumário

Lista de Figuras	ix
Lista de Tabelas	xi
1 Introdução	1
2 Ferramentas Usadas	3
2.1 Espectrograma	4
2.2 Separação espectral baseada em STFT	6
2.3 Da representação no tempo-frequência a representação no domínio tempo	7
2.4 Método de separação pela IBM	7
2.5 Coeficientes cepstrais da frequência Mel - MFCCs	10
2.6 Similaridade de Cossenos	16
3 Redes Neurais	17
3.1 CNN	17
3.2 U-Net	19
3.2.1 Convolução Transposta	21
4 Sobre os dados	24
4.1 Criação do Banco de Dados	24
5 Método Propostos	25
5.1 Modelo baseado na CNN	25
5.2 Modelo Baseado na U-Net	32
6 Conclusões	39
Referências Bibliográficas	40

Lista de Figuras

2.1	Representação de um modelo genérico de separação de áudio que utiliza o espectrograma	4
2.2	Ondas de áudio de 512Hz, 1024Hz e 2048Hz, misturadas artificialmente	4
2.3	Representação do espectrograma da onda de áudio da figura 2.2 . . .	5
2.4	Espectrograma ideal	6
2.5	Separação do áudio representado pela figura 2.2 usando o método da IBM	8
2.6	Separação vogal e ruído usando o método da IBM	9
2.7	Escala mel	10
2.8	Extraindo 39 MFCC de uma onda sonora digital	11
2.9	Exemplo de aplicação da janela Hamming	12
2.10	Banco de filtros na escala mel	13
2.11	Exemplo de cálculo do cepstrum	15
2.12	MFCC de um áudio	15
3.1	Exemplo de convolução entre o input e o kernel	18
3.2	Exemplo de max-pooling usando o filtro 2x2 e stride 2	19
3.3	Exemplo de segmentação de imagem	20
3.4	Imagem do modelo original da U-Net	20
3.5	Exemplo do método de convolução	21
3.6	Método de convolução não tradicional	22
3.7	Convolução apresentada como multiplicação de matrizes	22
3.8	Exemplo de convolução transposta	23
5.1	Mistura de áudios representadas no domínio tempo	25
5.2	Figura mostrando o MFCC do áudio atrás e na frente a forma de onda do áudio	26
5.3	Curva de aprendizado dependendo do ruído por trás	28
5.4	Classificação de vogais usando outros ruídos para treino	29
5.5	Descobrimo o instante aproximado em que a vogal foi dita usando uma análise pelo MFCC	29

5.6	Descobrimo o instante aproximado em que a vogal foi dita usando similaridade de cosseno no MFCC	30
5.7	A imagem do lado esquerdo representa a imagem salva do espectrograma e a imagem do lado direito a sua respectiva máscara binária	33
5.8	A imagem da esquerda mostra o caso em que não existe nenhum ponto interceptando as duas máscaras, a figura do meio quando parte das máscaras se interceptam e a da direita quando elas são exatamente iguais.	34
5.9	Otimizador: sgd, função de perda: sparse categorical cross entropy, na última camada a ativação é softmax, resultado nos dados de treino: perda: 0.032, e iou: 0.1241	35
5.10	otimizador: Adam, função de perda: binary cross entropy, na última camada ativação: sigmoid, perda: 0.5865 - iou_score: 0.478, epochs : 100	36
5.11	Curva de aprendizado da U-Net usando múltiplos ruídos juntos . . .	37
5.12	Dados de teste e seus respectivos resultados, na extrema esquerda, o espectrograma do áudio + ruído, no meio a máscara real e na direita á máscara gerada	38

Lista de Tabelas

5.1	Resultados obtidos usando as vogais puras, com a menor quantidade de ruído possível para testar se o método funcionaria para a classificação de áudios.	27
5.2	Tabela mostrando o resultado do tempo original de início de término da vogal, comparado com o obtido a partir do modelo proposto . . .	32
5.3	Resultados obtidos usando as vogais puras para treino e dados com ruído para teste	33
5.4	Resultados obtidos usando dados de treino e validação com diferentes tipos de ruídos diferentes	34

Capítulo 1

Introdução

É comum a situação na qual se deseja escutar música em presença de barulhos externos que dificultam essa escuta. Por exemplo, quando ao escutar o rádio no carro, ocorre a passagem de uma ambulância, a presença de barulhos de motor ou mesmo outros passageiros conversando em voz alta. Em ambientes assim, consegue-se ouvir vários sons diferentes vindo de fontes sonoras completamente distintas e usualmente distingui-los com facilidade. Além dessa capacidade de distinguir os sons, o motorista consegue focar em somente um dos sons e trocar esse foco como bem desejar, por exemplo, numa roda com diferentes pessoas e conversas paralelas, consegue-se em geral focar e participar de uma das conversas ignorando completamente as demais. Soluções para esse tipo de problema podem trazer aprimoramentos na diminuição dos ruídos em aparelhos auditivos, reconhecimento de fala, fala automática, entre outros problemas relacionados aos áudios.

Essas situações que acabamos de citar podem ser analisadas por diferentes vertentes, uma delas é conhecida como o ‘problema do coquetel’[1], se refere à habilidade em focar somente uma das fontes sonoras enquanto se ignora todas as demais, a outra se refere à capacidade de distinguir quais sons estão acontecendo simultaneamente, problema conhecido como segregação/separação dos sons recebidos ou auditory scene analysis(ASA)[2–5] . Nesta dissertação focaremos principalmente nesta segunda vertente.

A separação de sons possui alguns desafios, algumas situações impedirão de chegar ao som sem ruídos, como os caso em que se a quantidade de ruído presente em uma faixa de áudio for muito alta, ele irá mascarar o som almejado, esse problema pode ser observado quando uma ambulância com a sirene ligada se encontra bem próxima de um ouvinte, tornando muito difícil a escuta de outros sons, outra dificuldade é quando dois interlocutores que possuem timbres vocais muito similares falam simultaneamente, haverá dificuldade de distinguir as falas.

Para separar fontes sonoras, não é obrigatório descobrir ou classificar o tipo de som, mas é uma das abordagens que o trabalho explora. Alguns trabalhos estão mais

direcionados a descobrir alguns atributos dos sons como o tom e frame para a realizar a separação[3, 6, 7], já outros se utilizam de algumas métricas mais tradicionais como coeficientes cepstrais da frequência mel (MFCC)[8].

Em computacional auditory scene analysis(CASA) é bastante usual utilizar uma representação no tempo-frequência(TF) para gerar um filtro que ajuda na separação dos sons. Uma das representa

ções mais utilizadas para o som em TF, é o espectrograma, já que acredita-se que esse modelo gera um resultado que se assemelha à mecânica de como entendemos o som no nosso cérebro[3], esse modelo gera uma espécie de mapeamento do som no tempo, onde a imagem gerada é lida da esquerda para a direita(tempo), e na vertical estão representadas as frequências e suas respectivas magnitudes naquele intervalo de tempo.

Para encontrar o filtro para realizar a separação de sons, existem alguns métodos difundidos na área, e.g., é possível encontrar uma máscara binária ideal (IBM), essa máscara procura definir se a frequência pertence ou não ao som almejado, uma desvantagem deste algoritmo é a impossibilidade de separar sons que possuem frequências muito similares, como por exemplo duas pessoas de tom de voz similar falando simultaneamente. Outra forma de filtro é a máscara de proporção ideal (IRM), com esse método já é possível uma separação mais precisa entre duas fontes e também abrange os casos que os sons presentes são parecidos, pois ela decide a proporção de frequência que pertence a cada um dos sinais na mistura. Essas citadas anteriormente são as máscaras mais usadas na literatura, mas existem outras máscaras e outras abordagens para a separação de sons [9, 10].

Alguns estudos exploram outras maneiras de separação que incluem a análise de componentes independentes (ICA) e/ou fatorização de matrizes não negativas no domínio tempo (NMF), métodos mais usados quando se trabalha com uma separação cega de fontes(blind source separation BSS)[11], são modelos usado ao tentar fazer a separação das misturas sem nenhuma informação prévia do sinal, os métodos que usam a representação no TF tem como vantagem sua escalabilidade e possibilidade de generalização para uma quantidade grande de dados[12].

Como para este trabalho coletamos alguns dados para realizar os experimentos, temos o sinal puro e o sinal com o ruído adicionado artificialmente o que permite trabalhar com métodos de aprendizado supervisionado, por isso métodos de inteligência artificial, como CNN e UNeT auxiliados pelo MFCC e IBM serão explorados ao longo do texto.

Capítulo 2

Ferramentas Usadas

Os métodos propostos utilizam a representação TF do áudio, para isso foi escolhido o espectrograma e o MFCC para auxiliar. Uma das vantagens de usar o MFCC é o ganho na velocidade de classificação dos áudios, já que ocorre a diminuição da dimensão necessária para isso, já com o espectrograma consegue-se encontrar a IBM de uma maneira mais fácil.

Como ambas as ferramentas que estamos usando podem ser abordadas como se fossem imagens, será explicado futuramente o porquê, existe a possibilidade de trabalhar com modelos de redes neurais muito difundidos na área por fazerem um excelente trabalho em classificação e segmentação de imagens como a CNN e com a U-Net.

O espectrograma é uma espécie de mapa de calor das frequências do som representado por ele. Como ele está presente no domínio TF, é possível visualizar a magnitude das frequências em relação ao tempo, conseqüentemente uma quantidade maior de frequência num determinado intervalo de tempo se torna mais aparente na sua representação. Além do espectrograma, outra métrica que poderia ter sido utilizada é o cochleograma, porém não entraremos em detalhes em relação a esse método neste trabalho.

Os métodos mais populares de segmentação de sons são os que trabalham com o espectrograma para a sua separação[9]. A figura 2.0.0, mostra como alguns algoritmos de separação são executados. Nela, a saída é o áudio desejado separado do ruído. O ruído é definido como qualquer sinal não desejado, então mesmo que o ruído ao final dos algoritmos pareça ser um som com conteúdo, ainda terá essa classificação, por exemplo, se estivéssemos tentando separar uma parte instrumental de uma música da letra cantada, e a letra cantada fosse o objetivo final, o ruído seria a parte instrumental.

No esquema da 2.1, a entrada do algoritmo é um áudio ruidoso na representação no domínio do tempo, o algoritmo transforma essa mistura em uma representação TF, onde irá calcular a IBM relativa ao áudio de interesse. Como na IBM não

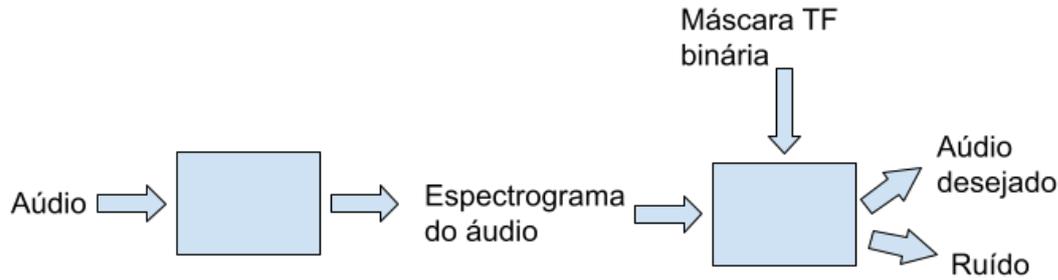


Figura 2.1: Representação de um modelo genérico de separação de áudio que utiliza o espectrograma

conseguimos separar as proporções relativas a cada um dos áudios, iremos assumir que os áudios são separáveis e não contêm muitas frequências comuns.

2.1 Espectrograma

Para reproduzir uma onda sonora de uma maneira perfeita artificialmente seria necessário uma quantidade infinita de amostras e a fim de decompô-la, uma quantidade infinita de frequências, porém como o computador tem uma capacidade limitada, essa representação é feita de uma maneira discreta, limitando assim a quantidade de frequências usadas na decomposição. A 2.2, mostra um exemplo de uma mistura artificial no domínio do tempo de três ondas de mesma amplitude, mas com frequências distintas, 512Hz, 1024Hz e 2048Hz.

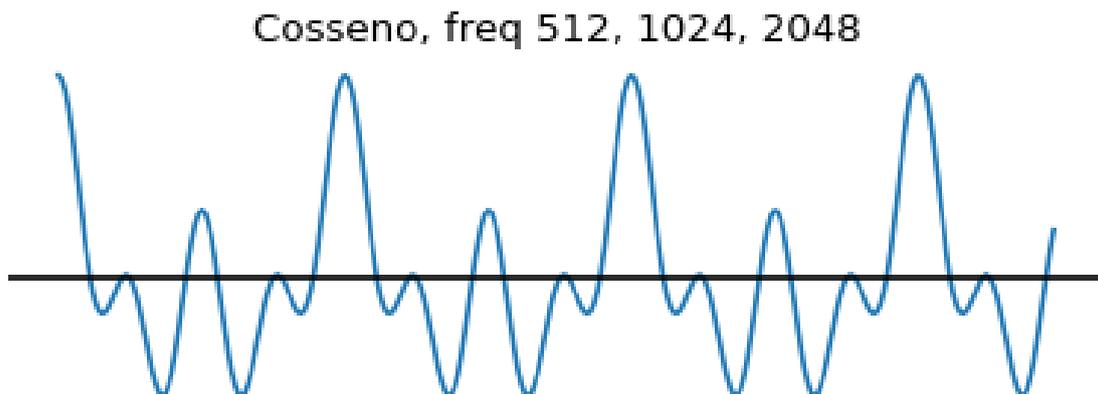


Figura 2.2: Ondas de áudio de 512Hz, 1024Hz e 2048Hz, misturadas artificialmente

A 2.2 mostra o áudio no domínio do tempo, com essa representação a amplitude da onda em relação ao tempo fica facilmente visível, porém é muito mais difícil saber as frequências presentes na onda. Uma boa maneira de contornar esse problema é usar uma representação no tempo-frequência. Esse trabalho optou pelo

espectrograma[13].

A 2.3, representa o espectrograma do áudio citado anteriormente, diferentemente da figura 2.2, ao olhar a imagem as frequências presentes na onda sonora 512Hz, 1024Hz e 2048Hz, estão mais bem destacadas que as outras frequências possíveis. Isso é uma característica do espectrograma, já que ele salva o valor da magnitude da onda em relação a um intervalo de frequência e tempo estabelecidos, essa representação se assemelha bastante a um mapa de calor, onde quanto mais “quente” está, maior é a influência da frequência naquele intervalo de tempo.

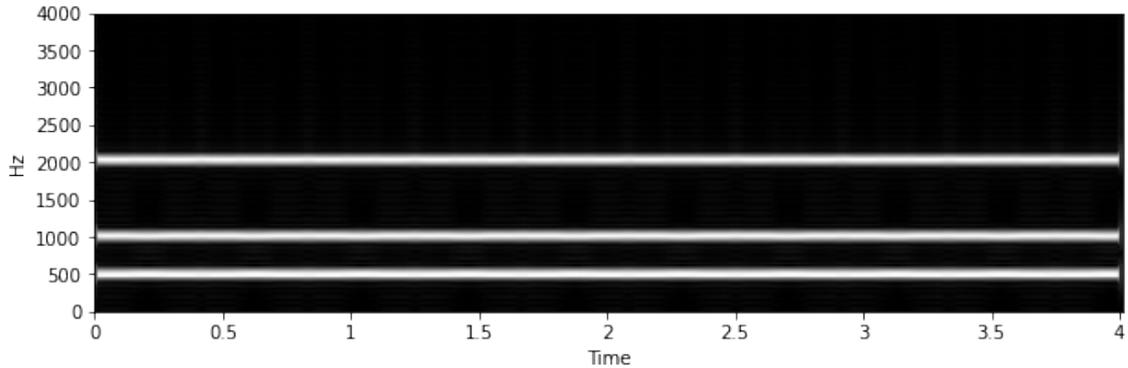


Figura 2.3: Representação do espectrograma da onda de áudio da figura 2.2

O espectrograma é calculado a partir da transformada de fourier no tempo curto (STFT), que é dada pela seguinte equação:

$$S(w, t) = F\{s(t)w(\tau - t)\} \quad (2.1)$$

Onde $s(t)$ é o sinal, $F\{\}$, representa a transformada de fourier relativa a $\phi(t)$ para $\phi(\omega)$ e $w(t)$ é uma função de peso aplicada ao sinal antes da transformada de fourier [14]. Após o sinal passar pela STFT, há a representação dele como parte real e parte imaginária, mas como o espectrograma é dado somente pela magnitude da transformada do áudio, não nos preocuparemos com a parte imaginária por enquanto.

$$espectrograma = |S(w, t)|^2 \quad (2.2)$$

Essa representação no tempo frequência é limitada, já que existe um limite máximo de intervalo de tempo/frequência. Cada “pixel” da imagem acima representa um intervalo de tempo e frequência que não podem ser diminuídos o quanto for desejado, já que para o cálculo do STFT existe um limite da janela intervalar, dado por $\Delta_t \Delta_\omega \geq 12$, onde Δ_t , tamanho da janela tempo, Δ_ω , tamanho da janela em relação à frequência, por isso não é possível aumentar a precisão de um dos eixos sem perder precisão no outro, o que é uma desvantagem da abordagem, pois no

domínio do tempo sempre há a possibilidade aumentar a quantidade de amostras. Então uma imagem que era para se assemelhar a figura 2.1.2, será na verdade a 2.3.

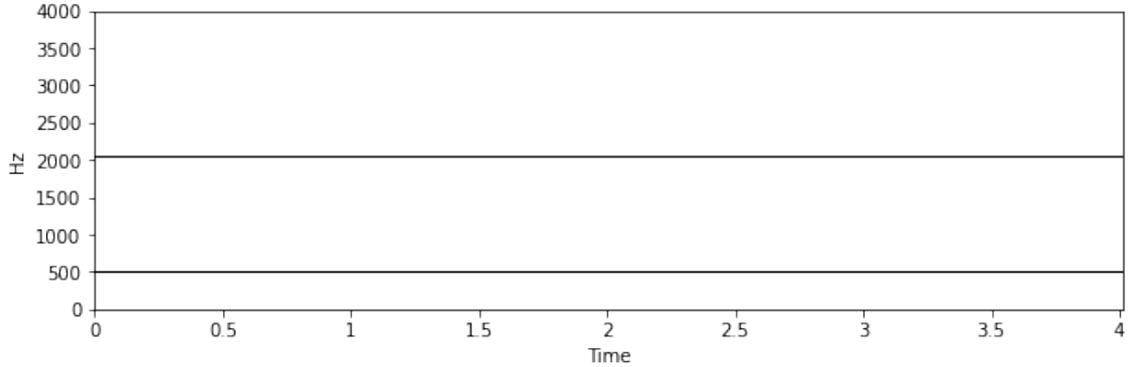


Figura 2.4: Espectrograma ideal

2.2 Separação espectral baseada em STFT

A subtração espectral é um método que pode ser usado para separar fontes sonoras. É um método não linear, que atua como um filtro, diminuindo e/ou eliminando uma das componentes presentes no sinal. O modelo inicial pode ser descrito pela equação abaixo:

$$s(t) = s_1(t) + s_2(t) \quad (2.3)$$

s é o áudio resultante da mistura dos áudios, s_1 é um dos áudios originais e s_2 é o outro áudio, a título de simplificação, seja s_1 e s_2 áudios que não são misturas e s_1 e s_2 estatisticamente independentes, ou seja, somente com a informação de quem é s_2 não resulta em alguma informação do áudio s_1 [15]. Como os sinais são estatisticamente independentes, ao realizar a transformada de Fourier de $s(t)$:

$$S(\omega) = S_1(\omega) + S_2(\omega) \quad (2.4)$$

Temos:

$$|S(\omega)|^2 = |S_1(\omega)|^2 + |S_2(\omega)|^2 \quad (2.5)$$

Se S_1 ou S_2 for estimável, então é possível fazer uma estimação do outro áudio. Por exemplo, imagine que \tilde{S}_2 seja uma boa estimativa da magnitude do áudio, então para uma boa aproximação de S_1 :

$$|\tilde{S}_1(\omega)|^2 = |S(\omega)|^2 - |\tilde{S}_2(\omega)|^2 \quad (2.6)$$

Onde S_1 é a estimação do áudio S_1 baseado em S_2 . O método de subtração espectral só modifica a magnitude dos áudios da mistura, deixando a fase deles inalterada, parte imaginária da transformada. Quando há a junção dos sinais, não há grandes modificações em suas fases, mas ao não se preocupar com elas, a separação pode ter um ótimo local. [12, 16] se preocupam em driblar esse problema, que não será abordado neste trabalho.

2.3 Da representação no tempo-frequência a representação no domínio tempo

Dado que $|\tilde{S}_1(\omega)|^2$ é o espectrograma relativo a S_1 , para representar esse sinal do domínio TF, é necessário as fases do sinal, como já foi discutido, não há grandes diferenças nas fases de S e \tilde{S}_1 , logo:

$$|\tilde{S}_1(\omega)| = |\tilde{S}_1(\omega)|^2 \angle S(\omega) \quad (2.7)$$

\tilde{S}_1 agora é uma representação no domínio TF, para retornar ao domínio do tempo, usa-se a transformada inversa de fourier no tempo curto (iSTFT), que é dada pela equação abaixo:

$$s_1(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} S(\tau, \omega) g(t - \tau) e^{ij\omega} d\tau d\omega \quad (2.8)$$

Onde $s_1(t)$ é o áudio que se quer recuperar e $S(\tau, \omega)$ é o sinal no TF, para mais detalhes da iSTFT pode-se consultar [13].

2.4 Método de separação pela IBM

Como foi explicado em 2.2, para realizar a separação espectral, é necessário que exista uma estimativa do espectro de uma das fontes sonoras a serem separadas e só então é feita a subtração espectral. O modelo da máscara binária ideal é quase o mesmo princípio, mas ao invés de subtrair um espectro do outro, o modelo escolhe quais são frequências que estão ou não presentes em um dos áudios, se presente, mantém a frequência com a mesma magnitude, se não, ela ignora a existência frequência, por isso a importância dos sinais não terem muitas frequências parecidas, pois se tiverem, esse método não gerará bons resultados.

Imagine que desejamos separar a onda de frequência 1024Hz que está presente na mistura de áudio representada pela figura 2.2 usando o método da IBM. O método da máscara binária ideal gerará uma máscara que eliminará todas as frequências que não pertencem ao som desejado, colocando o valor da magnitude como sendo 0 se

julgado que a frequência não pertence ao som e 1 caso contrário, após isso é feita a conversão da representação no domínio TF para o domínio tempo. A figura 2.4.0, ilustra um pouco como o método funciona.

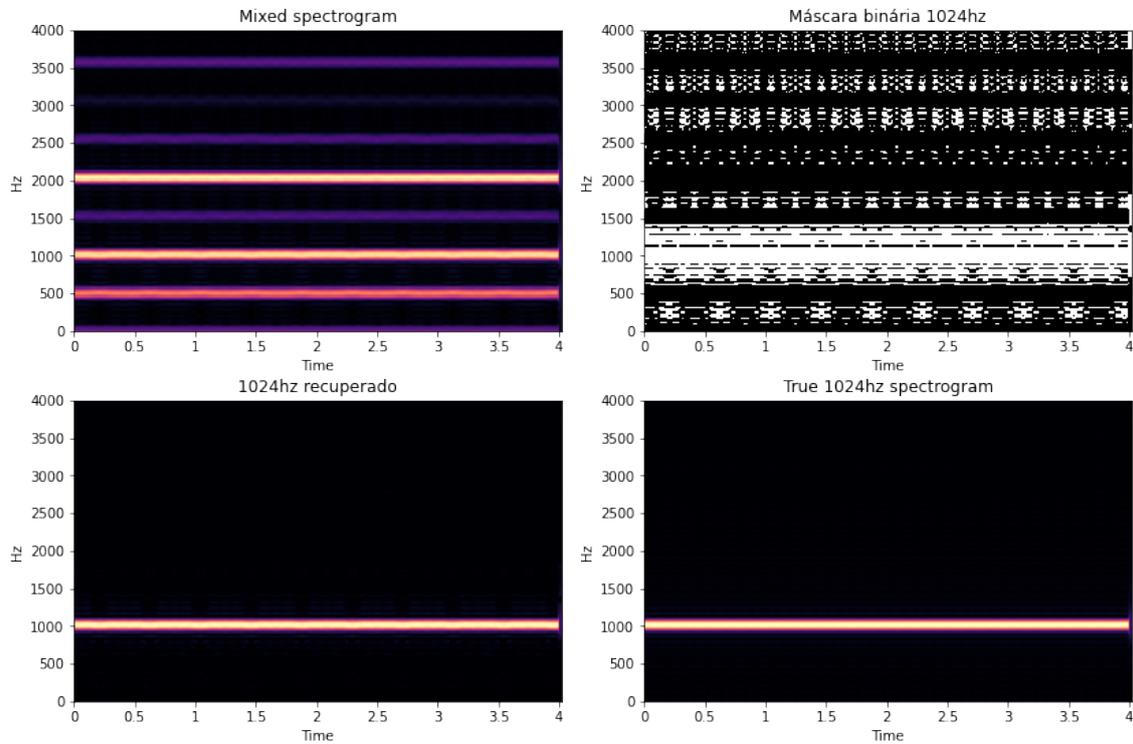


Figura 2.5: Separação do áudio representado pela figura 2.2 usando o método da IBM

Canto superior esquerdo: espectrograma de todas as ondas misturadas, canto superior direito: máscara binária obtida a partir do espectrograma de 1024hz, canto inferior esquerdo: espectrograma depois que utilizamos a máscara binária para separar as ondas, canto inferior direito: espectrograma da onda de 1024hz original.

A imagem no canto superior esquerdo da figura 2.5 representa o espectrograma da mistura, já do canto superior direito é representação da máscara binária calculada a partir do espectrograma original da onda de 1024Hz, a figura do canto inferior esquerdo representa o espectrograma resultante da aplicação da máscara binária no espectrograma da mistura. É possível ver que o espectrograma resultante é bem similar ao espectrograma da onda original, imagem no canto inferior direito. O resultado obtido é redundante, dado que já existe o espectrograma dos áudios separados, mas sua função foi auxiliar didaticamente a explicação do método e sua eficácia. A figura 2.4.1, apresenta a aplicação do método num som mais complexo que a mistura de ondas.

No canto superior esquerdo da figura 2.6 está representado o espectrograma de uma vogal falada misturada artificialmente a um trecho de uma música instrumental de fundo. Para os seres humanos é fácil identificar a região aproximada da vogal,

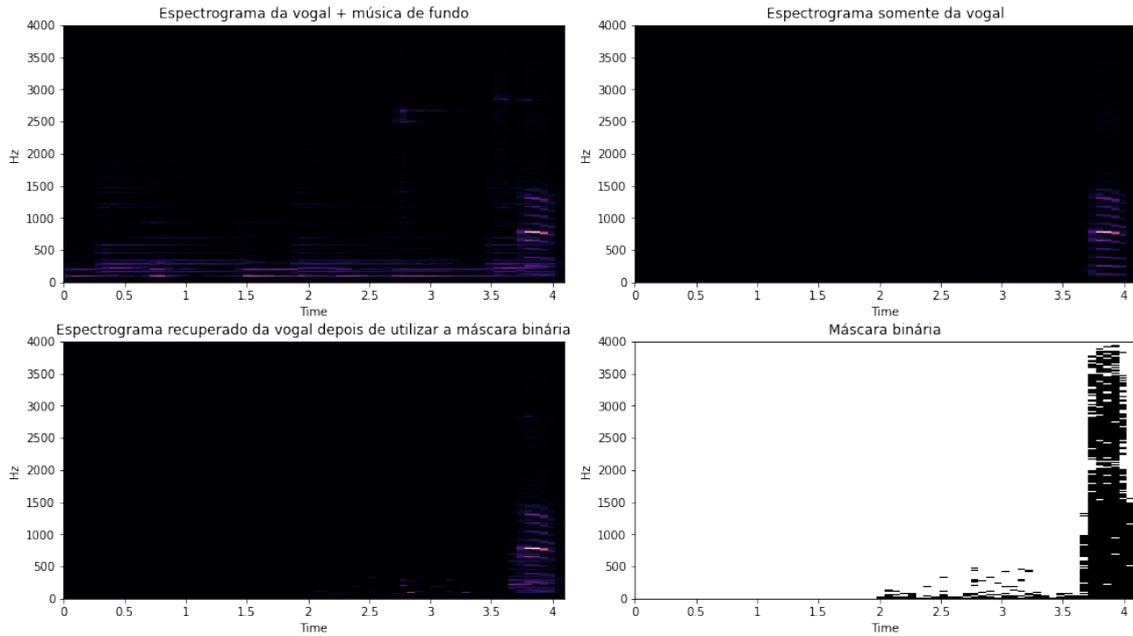


Figura 2.6: Separação vogal e ruído usando o método da IBM

Canto superior esquerdo: representação espectral do áudio da mistura artificial ruído + vogal, canto superior direito, imagem do espectrograma da vogal pura, canto inferior direito, imagem da máscara binária obtida a partir do espectrograma da vogal a, canto inferior esquerdo, espectrograma da vogal a, recuperada a partir da máscara binária

dado que ela é ligeiramente diferente do restante da imagem, mas é difícil separar quais frequências pertencem a uns dos sons e quais pertencem ao outro, tornando-se uma tarefa difícil tanto para humanos quanto para o computador, por isso foi estimado uma máscara, canto inferior direito da figura 2.6, baseada no espectrograma original representado pela imagem no canto superior direito da mesma figura, o resultado está ilustrado no canto inferior esquerdo da mesma figura. É possível observar que nessa nova situação a separação não ficou tão próxima da original quanto a do experimento mostrado na 2.5.

A máscara binária é gerada a partir do áudio de entrada, seu cálculo é dado pela seguinte equação:

$$IBM(t, f) = \begin{cases} 1, & SRN(t, f) > LC \\ 0, & \text{c.c} \end{cases} \quad (2.9)$$

Essa equação procura encontrar no espectrograma qual frequência pertence ao sinal e qual pertence a um ruído. Signal-to-noise ratio(SRN) é uma função que tem como objetivo comparar a proporção de sinal que está presente na mistura ruído+sinal, se num dado tempo-frequência ela for maior que um critério local, então esse ponto é predominantemente sinal, ainda podendo ter algum ruído restante, já se esse valor for predominantemente ruído, então nesse caso não vale a pena conservar

essa informação, por isso nesse local a máscara recebe 0[17].

2.5 Coeficientes cepstrais da frequência Mel - MFCCs

Os MFCCs foram introduzidos por Davis e Mermelstein em 1980, são muito usados na literatura para reconhecimento de som e fala automática. Sua base teórica é formada pelo cepstrum juntamente com a escala mel. Cepstrum, início da palavra spectrum invertido, é o espectro do logaritmo do espectro de uma forma de onda[18], já a escala mel, 2.7 é uma escala perceptiva, que mantém a relação de percepção humana dos sons, ou seja, os tons julgados com intervalos equiespaçados por parte de observadores, continuam equiespaçados nessa escala.

Para conseguirmos extrair informações do MFCC, o primeiro passo é conseguir transformar um áudio do mundo real para um áudio representado computacionalmente. Para passar do analógico para o digital o áudio passará por dois processos: amostragem e quantização.

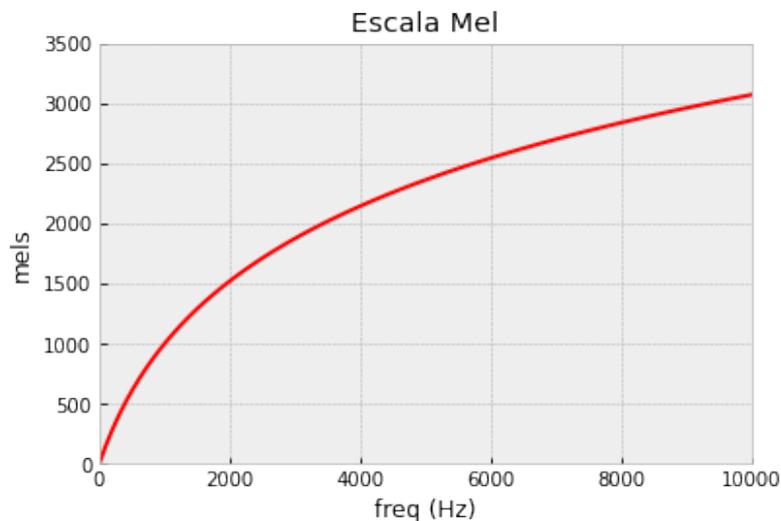


Figura 2.7: Escala mel

Amostragem é a quantidade de amostras do áudio que conseguimos ter informações durante um segundo, por exemplo ao usar um amostragem de 10kHz, estamos pegando 10000 amostras do áudio em um segundo, onde cada amostra contém o valor da intensidade do sinal naquele dado instante de tempo, gerando um problema numérico já que essa intensidade é um número real, analógico e para passar para o digital, precisamos usar a quantização, passar esse valor real para um valor inteiro que o computador possa representar, geralmente utiliza-se um número de 8 ou 16 bits.

A frequência Nyquist nos diz qual o tamanho do sampleamento ideal para não perdermos nenhuma informação do áudio, ela diz que a máxima frequência que podemos ter numa onda digital é a metade do valor do sampleamento, ou seja, se a máxima frequência que queremos captar é 10KHz, temos que usar um valor de sampleamento de 20kHz. Com esse sinal digital, consegue-se extrair os MFCCs, a figura ?? mostra os passos necessários para essa operação.

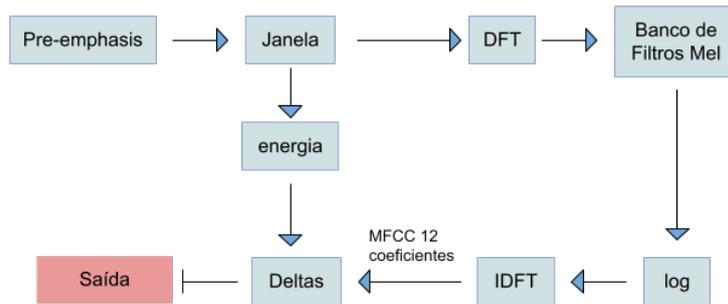


Figura 2.8: Extraindo 39 MFCC de uma onda sonora digital

O primeiro passo para gerar o MFCC é aplicar um filtro pré enfático a fim de amplificar as frequências mais altas, já que elas geralmente possuem menos energia, isso faz com que as informações desses níveis mais altos sejam mais disponíveis para o modelo.

Como o áudio geralmente é um sinal não estacionário, não periódico, subdivide-se o sinal de áudio em intervalos pequenos que garantam que naquele intervalo seu comportamento seja periódico, para isso pode-se utilizar janelas. Essas janelas têm valores não-negativos numa região específica e valores zerados fora dessa região, ao passar essa janela pelo sinal, é possível extrair uma parte do sinal dentro dessa janela. Existem alguns parâmetros importantes para escolher a janela, como seu tamanho em ms, o offset e o formato. Já o sinal extraído de cada uma das janelas é chamado de frame.

Para extrair o sinal, utilizamos a equação abaixo, onde $s[n]$ é o valor do sinal no tempo n e $w[n]$ é o valor da janela no tempo n :

$$y[n] = w[n]s[n] \quad (2.10)$$

Um dos tipos de janela muito utilizados é a janela Hamming. Ela ameniza o sinal perto das bordas, fazendo com que não haja descontinuidade no sinal extraído. Matematicamente ela é descrita pela equação 2.11 e um exemplo de aplicação está ilustrado na figura 2.5.2.

$$w[n] = \begin{cases} 0.54 - 0.46 \cos(2nL - 1), & \text{se } 0 \leq n \leq L - 1 \\ 0, & \text{caso contrário} \end{cases} \quad (2.11)$$

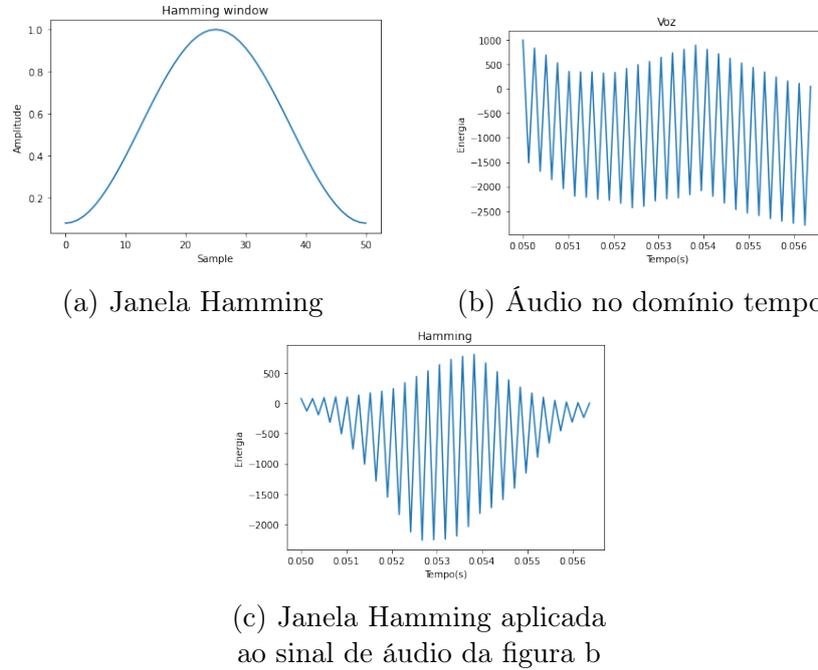


Figura 2.9: 2.9a: Aparência da janela hamming, de 50 amostras, 2.9b: Pedaco do sinal de áudio, 2.9c: Pedaco do sinal de áudio b, após passar pela janela Hamming, suas bordas atenuadas

Após a utilização das janelas, criando um intervalo estacionário, se torna viável aplicar a transformada discreta de fourier (DFT). Essas operações fazem com que seja possível obter a informação sobre a quantidade de frequência em cada intervalo de tempo do sinal. O próximo passo do algoritmo é a utilização da escala Mel nesses intervalos.

Os seres humanos não conseguem distinguir todas as frequências de maneira linear, é muito mais fácil saber que houve uma mudança na frequência quando estamos lidando com frequências mais baixas, do que reconhecer a mesma mudança em frequências mais altas. Por exemplo, facilmente percebemos que a frequência mudou de 500Hz para 600Hz, mas a mesma mudança não é perceptível com 15KHz e 15.1KHz. Para lidar com essa situação foi criada a escala Mel. Ela é representada pela seguinte equação:

$$mel(f) = 2595 \log_{10} \left(1 + \frac{1}{100} \right) \quad (2.12)$$

Um dos passos finais para o cálculo do MFCC é criar um banco de filtros, aplicando filtros triangulares, onde a quantidade de filtros é definida previamente, normalmente usam-se 40 filtros na escala Mel para extrair as frequências de banda.

Podemos converter entre Hertz(f) e Mel(m), usando a equação abaixo[19]:

$$f = 700(10^{\frac{m}{2595}} - 1) \quad (2.13)$$

Cada filtro dentro do banco de filtros é uma onda triangular, centrada numa frequência, cujo valor nesse ponto é 1 e depois decai linearmente até as frequências adjacentes, quando atinge o valor 0. Como ilustrado na figura ??.

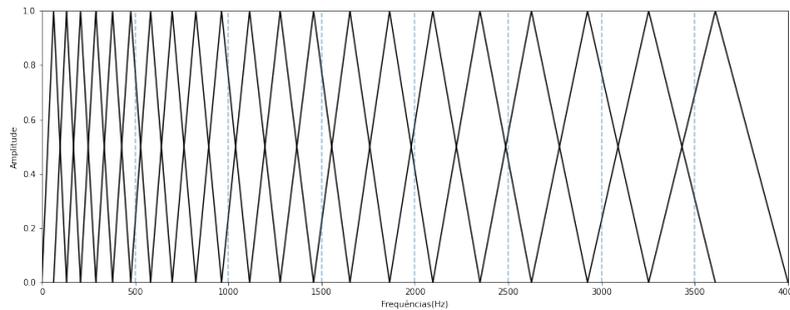


Figura 2.10: Banco de filtros na escala mel

Uma das maneiras de gerar esse banco de filtros é transformar a frequência inicial e a final para a escala Mel e separá-los linearmente na quantidade de filtros desejados, após isso voltamos essas frequências que foram separadas na escala Mel, para a escala Hz e com isso consegue-se as frequências necessárias para gerar os filtros triangulares.

O próximo passo do algoritmo é usar o cepstrum. O cepstrum pode ser interpretado como uma taxa de variação entre as frequências espectrais, ajudando a ver as frequências fundamentais do áudio. Para fins de simplificação de explicação, será calculado o cepstrum do áudio inicial e não o cepstrum da frequência mel (MFC), que é onde as frequências são linearmente espaçadas na escala mel, explicada anteriormente.

O cepstrum é o espectro do log do espectro. A primeira transformação faz com que o sinal saia do domínio temporal para o domínio espectral(frequência/magnitude), já o log, altera somente a magnitude do sinal e o espectro do log do espectro retornará o sinal no domínio espectral para o temporal, porém esse sinal resultante não é o mesmo sinal que entrou neste algoritmo. Essa nova representação auxilia a identificar as frequências mais importantes do sinal. A figura 2.11 ilustra essa mudança.

Um dos motivos para transformarmos o áudio para o cepstrum é que nesse domínio, os coeficientes cepstrais são descorrelacionados, o que não ocorre com os coeficientes espectrais. Geralmente para gerar o MFC, utiliza-se os 12 primeiros coeficientes cepstrais. De uma maneira formal pode-se dizer que o cepstrum é a

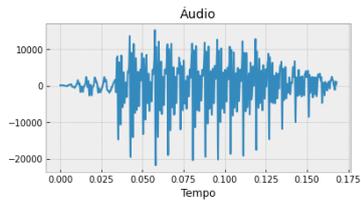
transformada discreta inversa de fourier da magnitude do log da transformada discreta de fourier do sinal.

Após extrair os 12 coeficientes cepstrais, é necessário adicionar outra informação, a energia de cada janela(frame), que é dada pela equação abaixo:

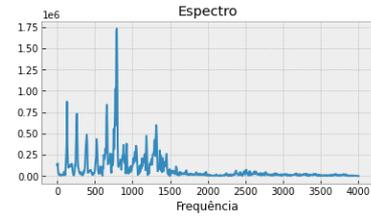
$$Energia = \sum_{t=t_1}^{t_2} x^2[t] \quad (2.14)$$

Onde t_1 é o tempo inicial da janela, t_2 é o tempo final, $x[t]$ é a amostra do sinal no tempo t . Como a energia do sinal não é constante ao longo do tempo, é necessário saber a mudança em cada janela correspondente ao coeficiente cepstral/energia, sua velocidade, e a mudança entre cada janela, a aceleração. Ao final do cálculo do MFCC, temos as seguintes informações:

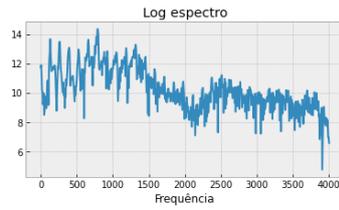
- 12 coeficientes cepstrais
- 12 coeficientes cepstrais de velocidade
- 12 coeficientes cepstrais de aceleração
- 1 coeficiente de energia
- 1 coeficiente de energia de velocidade



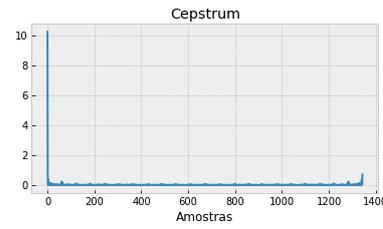
(a) Sinal na representação do tempo



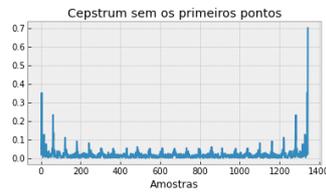
(b) Magnitude do espectro



(c) Log da magnitude do espectro



(d) Cepstrum



(e) Cepstrum sem os primeiros pontos

Figura 2.11

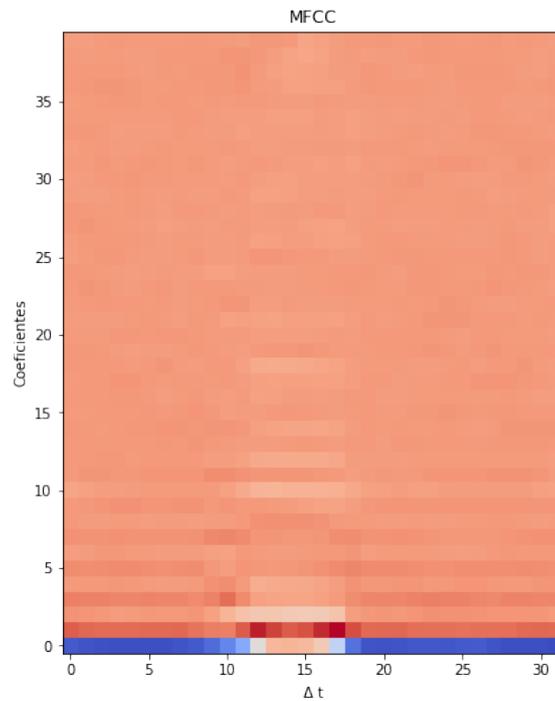


Figura 2.12: MFCC de um áudio

2.6 Similaridade de Cossenos

A similaridade de cossenos é uma métrica que permite calcular a distância entre dois vetores, o cálculo é dado pela equação abaixo. Quanto mais próximo de um o resultado, mais similar são os dois vetores

$$\text{cossim}(A, B) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^n A_i \cdot B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}} \quad (2.15)$$

Capítulo 3

Redes Neurais

Nesse capítulo será apresentado de forma sucinta os modelos de redes neurais utilizados nesta dissertação. É possível observar que a separação dos sons pode ser feita somente usando a “imagem” dele, por isso duas redes neurais bastante populares quando se trabalha com imagens são a redes neurais convolucionais(CNN) e a U-Net.

3.1 CNN

A rede neural convolucional [20] é um tipo de rede neural artificial que tem como característica a capacidade de aprendizado dos filtros necessários para a classificação, por isso é um modelo muito utilizado na classificação de imagens e no processamento de vídeos[21]. Seu nome se deve ao fato de utilizar a convolução como ferramenta para auxiliar o aprendizado, essa ideia ajuda a resolver alguns desafios do aprendizado de máquina, como interações esparsas, compartilhamento de parâmetros e representações equivariantes.

A convolução na matemática é dada pela seguinte equação:

$$s(t) = (x * \omega)(t) = \int x(a)\omega(t - a)da \quad (3.1)$$

Onde x e ω são funções definidas no domínio dos números reais. Como a CNN é um modelo computacional, é utilizado a convolução discreta ao invés da contínua.

$$s(t) = \sum_{a=-\text{inf}}^{\text{inf}} x(a) * \omega(t - a) \quad (3.2)$$

Ao usar a CNN na classificação de imagens, a entrada da rede são imagens, que na equação acima são representadas pela função x . Uma imagem podem ser representada computacionalmente de algumas maneiras, a primeira é utilizando três matrizes onde cada uma representa a quantidade de verde, vermelho e azul presente

em cada pixel (representação RGB), outra possibilidade é adicionar outra matriz ao modelo anterior para representar a opacidade da imagem (RGBA) e caso a imagem só contiver variações do branco ao preto, pode ser utilizado grayscale.

Como a maioria das imagens utiliza o modelo de RGB, então nesse caso x é um tensor de três dimensões, e a função \cdot , será o kernel (filtro), que ajuda a rede a aprender sobre a estrutura espacial da imagem, esse filtro como citado anteriormente será aprendido no modelo. A CNN na maioria das aplicações usa a convolução junto com outras funções, definiremos essas operações como sendo uma camada da rede.

Assim que a CNN recebe como entrada uma imagem, é feita a convolução dela com o kernel que gera um resultado, também conhecido como mapa de recursos, esse output possui como tamanho final, o tamanho inicial da imagem a menos do tamanho do kernel mais um pixel, por exemplo, na figura 3.2 o tamanho da matriz é 5×5 , já o do kernel é 2×2 , então o resultado final tem o tamanho do kernel a menos $(5 \times 5 - 2 \times 2) = (3 \times 3)$, mais um pixel nas dimensões $(3 \times 3) + (1 \times 1) = (4 \times 4)$.

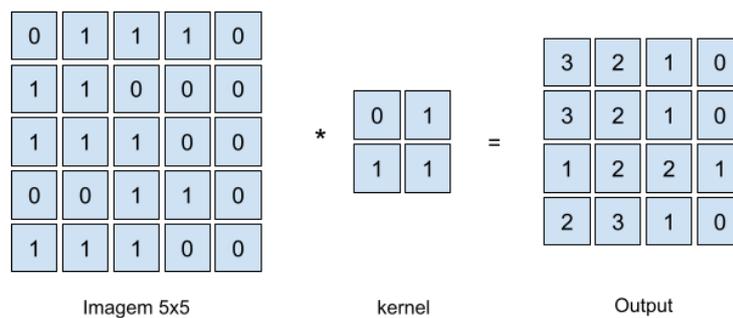


Figura 3.1: Exemplo de convolução entre o input e o kernel

Caso o kernel seja de um tamanho diferente da imagem, ou seja, ao realizar a convolução não é possível “aplicar” um número inteiro dos mesmos kernels para realizar a convolução, é possível solucionar esse problema, usando o padding que consiste em adicionar colunas de zeros nas bordas dos inputs, outra solução possível seria eliminar algumas das bordas presentes no input.

Após essa etapa de convolução, usualmente coloca-se uma função para adicionar não linearidade à rede, essa função recebe o nome de função de ativação. Uma das funções mais usadas é a ReLU (unidade retificada linear):

$$f(x) = \max(0, x) \tag{3.3}$$

A função sigmoideal também pode ser usada para esse propósito:

$$s(x) = \frac{1}{1 + e^{-x}} \tag{3.4}$$

Após passar por essa função de ativação, existe a etapa de pooling, que reduz a quantidade de neurônios. Uma das maneiras de realizar essa operação é usando o pool máximo, Max-pooling, que gerará uma ativação máxima na região. A figura 3.1.1 ilustra como funciona um filtro 2x2, usando um stride de tamanho 2. Stride é a quantidade de pixels que são pulados após o uso do filtro, por exemplo, se o stride fosse de tamanho 1, o próximo filtro seria a matriz, $[[2,1],[2,1]]$, ao invés da $[[1,0],[1,0]]$, como é no caso do stride 2.

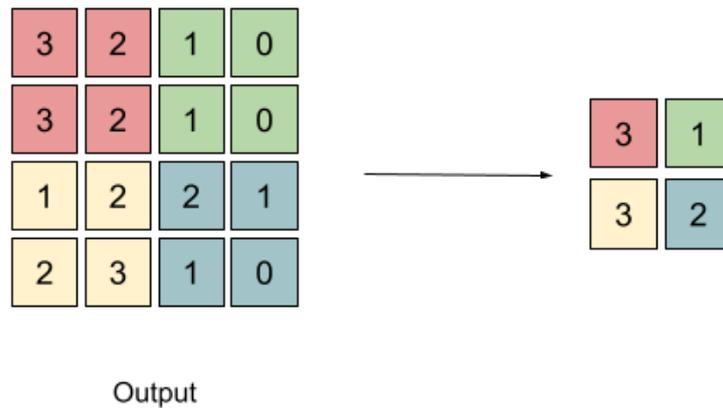


Figura 3.2: Exemplo de max-pooling usando o filtro 2x2 e stride 2

Essas funções em sequência definem uma camada da CNN. Normalmente é usado mais de uma camada na construção da rede e a quantidade de camadas usadas depende do tamanho das imagens de entrada e da quantidade de atributos para aprender. Após definir a quantidade dessas camadas, coloca-se uma última camada totalmente conectada, ou seja, conecta todos os neurônios de saída da última camada a cada um dos neurônios de saída da rede, completando assim o modelo da CNN.

3.2 U-Net

A U-Net é uma rede neural bastante usada para a segmentação de imagens. Segmentar imagens é um problema de visão computacional, que procura particionar a imagem em segmentos ou pixels. A figura 3.3, mostra um exemplo de uma possível segmentação, os pixels pretos são definidos como sendo background da imagem e os brancos o animal.

A U-Net é construída de uma maneira que um dos caminhos reduz a dimensão do input, assim como na CNN, esse caminho classifica cada um dos pixels, já o outro caminho expande a dimensão do output, para que depois que as regiões sejam devidamente classificadas, seja possível retornar à segmentação no mesmo tamanho que

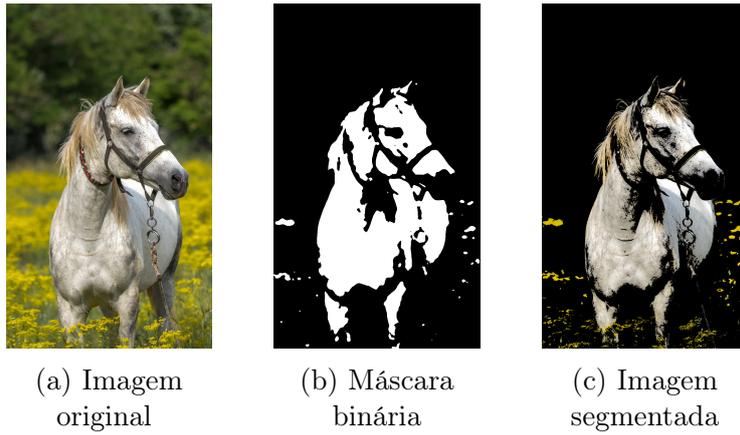


Figura 3.3

a imagem original. Esses caminhos são “paralelos”, figura ??, se assemelhando em formato à letra U. Esse caminho de upsampling, permite à rede propagar informações de contexto às camadas superiores e também faz com que a rede seja simétrica à parte que diminui a dimensão.

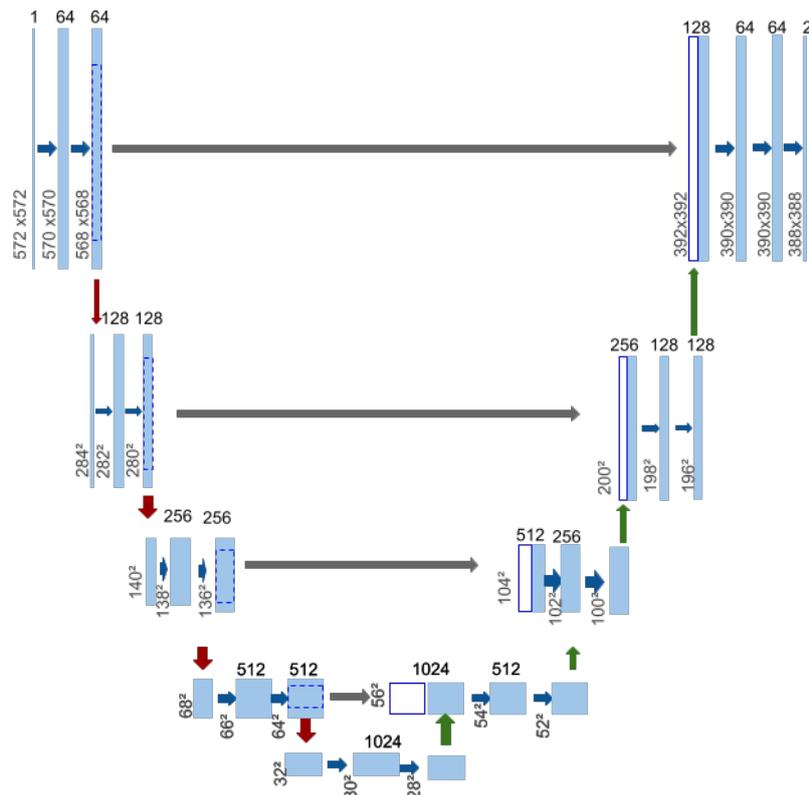


Figura 3.4: Imagem do modelo original da U-Net

A figura 3.4 ilustra a primeira U-Net modelada[22]. Ela tem uma imagem 572x572 como entrada e cada seta para azul a direita é uma operação de convolução seguida pelo ReLU, já a seta vermelha para baixo, representa uma operação de max pool 2x2, essa estrutura se assemelha bastante a uma CNN. As setas cinzas

representam a operação de corte e concatenação do tensor do caminho de downsampling ao de up-sampling, essa operação é importante, já que ao fazer a convolução perdemos os pixels das bordas, essa é uma maneira de recuperar essa informação. As setas verdes representam a convolução transposta 2x2 que diminui que diminui pela metade a quantidade de canais de atributos, seguida por uma concatenação com o mapa de atributos da caminho de compressão. A última camada do up-sampling é uma convolução 1x1 para mapear os pixels das classes desejadas. A rede possui 23 camadas convolucionais.

3.2.1 Convolução Transposta

Existem algumas estratégias possíveis para aumentar a resolução de uma imagem, como interpolação linear, quadrática, cúbica, sinc, entre outras. Esses algoritmos geram combinações lineares/não lineares de valores já existentes dos pixels, não aprendendo nenhuma informação nova, ao contrário do kernel na CNN, então se for desejado que o algoritmo aprenda como aumentar a resolução, a convolução transposta pode ser usada.

Será revisto o método da convolução para facilitar o entendimento da convolução transposta.

A figura 3.5 mostra um exemplo do método de convolução. A figura 3.6, ilustra uma maneira diferente de visualizar esse método. A imagem de 4x4 pode ser achatada de maneira que seja um único vetor 16x1, figura 3.6a, já o kernel 3x3 pode ser transformado numa matriz 4x16, figura 3.6b, se multiplicarmos essas matrizes, teremos o mesmo resultado que se tiéssemos feito a convolução com as matrizes originais, porém esse resultado é achatado, 4x1, só um vetor, para transformá-lo no resultado esperado, só passar para a forma da matriz 2x2, figura 3.7[23].

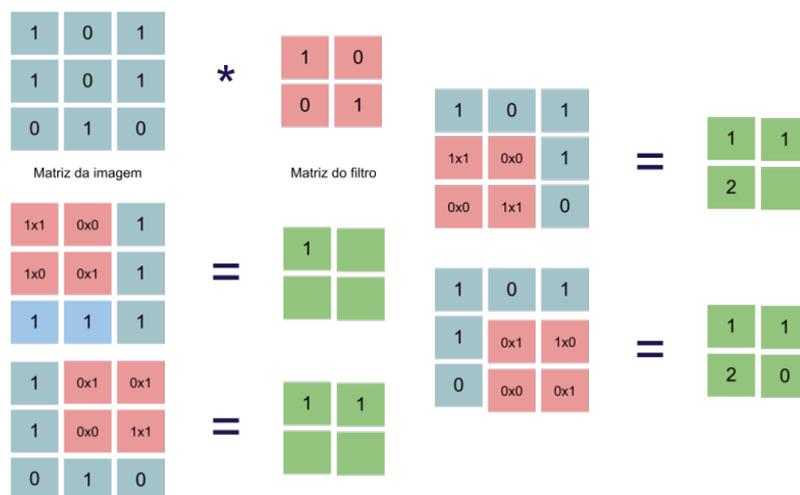


Figura 3.5: Exemplo do método de convolução

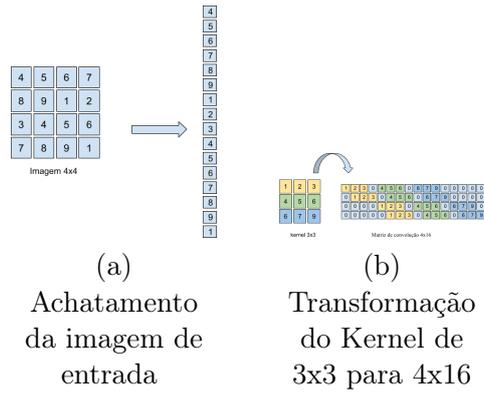


Figura 3.6

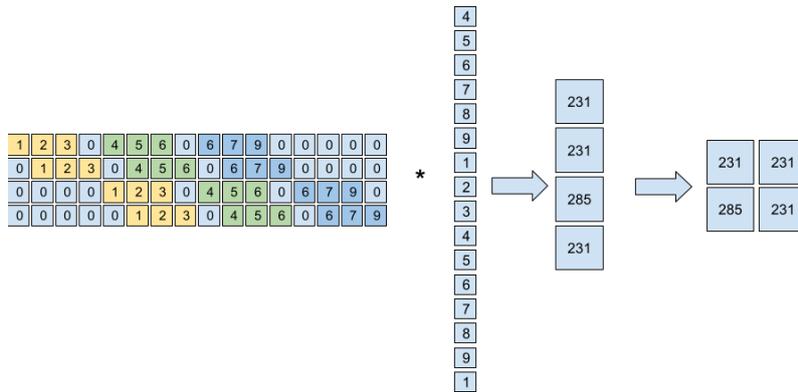


Figura 3.7: Convolução apresentada como multiplicação de matrizes

A operação da convolução resulta em uma redução de dimensionalidade, no exemplo acima, a matriz 4x4 (vetor 16x1), foi para 2x2(vetor 4x1), já a convolução transposta tem o objetivo oposto, sair do vetor 4x1(matriz 2x2) e chegar no 16x1(matriz 4x4). A operação a ser feita é transpor o kernel, de 16x4 passa a ser 4x16 e com esse kernel transposto, fazer a multiplicação pelo vetor 4x1(matriz 2x2), com isso é possível recuperar o vetor imagem 16x1 (matriz 4x4), figura ???. Esse kernel que foi transposto pode ser aprendido na convolução transposta, o que é uma vantagem para o método.

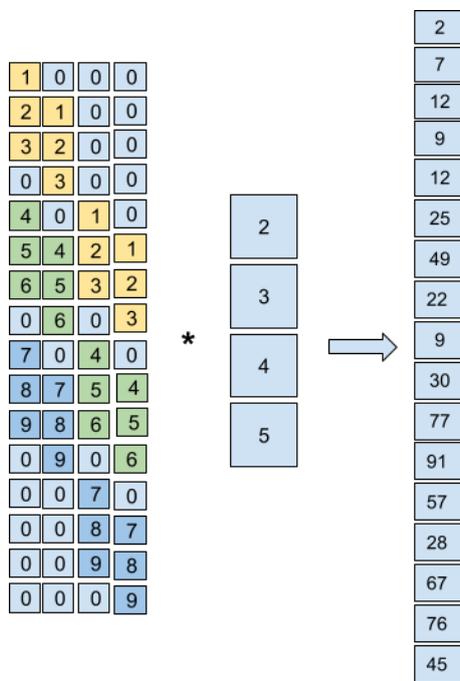


Figura 3.8: Exemplo de convolução transposta

Capítulo 4

Sobre os dados

Nos experimentos realizados usou-se dados criados artificialmente com o objetivo de gerar uma versão mais controlada e simplificada do problema, já que a separação de fontes sonoras abrange uma gama de subproblemas que dificultam o trabalho. Alguns deles são: a interferência de uma onda na outra caso os sinais tenham sido gravados no mesmo microfone e tenham vindo de fontes diferentes, o ruído presente no ambiente adicionando mais informação ao sinal, a reverberação dos sinais dependendo do ambiente no qual tenham sido gravados, muitas frequências comuns a ambos os sinais, dificultando ainda mais a separação, entre outros complicadores.

4.1 Criação do Banco de Dados

Os dados para esse experimento foram recolhidos a partir de 4 voluntários, dois homens e duas mulheres que gravaram as vogais a,e,i,o,u cerca de 20 vezes para cada vogal, totalizando em torno de 400 áudios. Os áudios foram gravados de maneira que todos possuem 2 segundos de duração, onde a maior parte de seu conteúdo é composta por silêncio e o mínimo de ruído externo possível. Os áudios gravados possuem um canal, áudios monofônicos, a frequência utilizada para gravá-los foi de 8kHz, um valor aparentemente baixo, mas suficiente para extrair as informações necessárias, dado que o som das vogais é muito simples e elas possuem frequências predominantemente baixas[18].

A partir desses áudios das vogais foram geradas misturas artificiais. Essas misturas contém esses sons previamente gravados com outros tipos de sinais, que também foram recortados para terem 2s de duração e 8kHz de frequência de amostragem. Foram usados sons de cachorro latindo, músicas instrumentais, o barulho da chuva caindo e o som de maritacas brigando.

Capítulo 5

Método Propostos

O problema abordado no trabalho é conseguir extrair a vogal dessa mistura artificial, para isso foram testados dois modelos, um usando a CNN, que não teve resultados satisfatórios, e outro usando a U-Net, cujos resultados observados foram melhores.

5.1 Modelo baseado na CNN

No primeiro modelo testado foi feita a classificação dos sons e após isso a sua separação, para essa tarefa as ferramentas escolhidas foram a CNN juntamente com o MFCC. A escolha de usar o MFCC ao invés do espectrograma que contém mais informação sobre os sons, foi para diminuir a dimensão do problema por selecionar somente alguns coeficientes por intervalo de tempo.

A ideia desse modelo surgiu após a observação dos gráficos representados pela figura ??, onde é possível notar que a influência do áudio instrumental sobre o áudio da vogal não é muito grande, então encontrando o intervalo de tempo que a vogal está presente já haveria uma separação bem rudimentar.

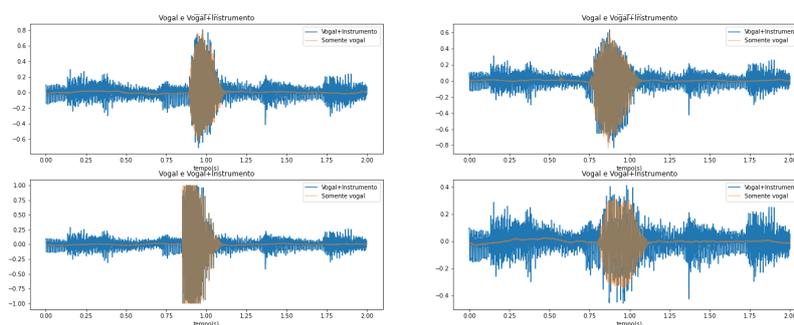


Figura 5.1: As figuras representam vários áudios diferentes misturados com o mesmo áudio instrumental, é possível observar que a onda no local misturado, onda em azul, não sofre grandes alterações em relação à forma de onda original, a onda em laranja.

Foi feito um pré-processamento dos dados, onde foram extraídos os MFCCs dos

áudios. A figura 5.12 mostra como o MFCC ajuda na classificação das vogais. Na figura 5.2a temos o MFCC do áudio + instrumental, nessa imagem é possível observar a relação entre o MFCC e o áudio no espaço tempo, já na 5.2b temos o mesmo áudio, sem a parte instrumental, nessa imagem a relação do MFCC e do áudio está mais aparente.

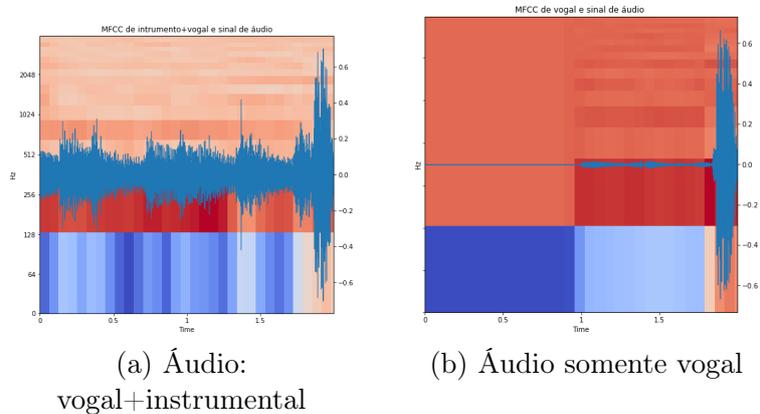


Figura 5.2: Representação em MFCC, background, junto com a representação do áudio no domínio tempo, onda em azul

O primeiro teste realizado foi conferir a acurácia da classificação dos áudios sem ruídos externos, áudios somente com a vogal, usando a combinação de CNN e MFCC. Os dados foram separados em 21% dados de validação, 9% dados de teste e 70% dados de treino separados de forma aleatória.

A entrada da rede recebe os MFCCs dos áudios com suas respectivas classificações, ‘a’, ‘e’, ‘i’, ‘o’, ‘u’. Os MFCCs foram gerados usando frequência de amostragem de 8kHz, a mesma utilizada para gravar os áudios, foram extraídos 40 coeficientes, com 512 amostras entre janelas sucessivas. A CNN tem quatro camadas, as três primeiras possuem a seguinte configuração: um layer de convolução de kernel 2x2, função de ativação ReLU e pool máximo de tamanho 2, com dropout de 20% para prevenir overfitting e a última camada teve como ativação o softmax.

O otimizador escolhido foi o ‘adam’, da biblioteca Tensorflow, a função de perda escolhida foi a função categórica ‘cross entropy’. A tabela 5.4 mostra o primeiro teste feito usando k fold de tamanho 5, rodando com 50 épocas e tamanho de batch 20. Esse primeiro resultado foi utilizando somente as vogais sem instrumento.

Para analisar o comportamento do método de MFCC+CNN, a rede foi treinada com as misturas de áudios, onde o áudio da classificação continuava sendo as vogais e o áudio por trás era sempre o mesmo ruído para cada banco de dados usado. Foram usadas diversas misturas, como chuva e vogal, latido e vogal. Podemos verificar pela tabela 5.4 que os resultados foram satisfatórios em relação a robustez do método. Intuitivamente faz sentido que a melhor classificação ocorra quando treinamos somente com vogais sem ruído o que os experimentos corroboraram, também

Kfold	Acertos dados de Teste	Acertos dados de treino
1	78%	88%
2	94%	92%
3	91%	88%
4	88%	89%
5	89%	90%

Tabela 5.1: Resultados obtidos usando as vogais puras, com a menor quantidade de ruído possível para testar se o método funcionaria para a classificação de áudios.

observamos que alguns dos ruídos não tiveram muita influência na classificação.

Podemos observar pelos gráficos da 5.3 que a curva de aprendizado se comporta de maneira muito semelhante usando qualquer um dos datasets.

Como funcionou relativamente bem o método de classificação usando o MFCC+CNN, tentaremos separar os ruídos por trás. Já que ao usar somente os dados puros para treino conseguiu-se uma acurácia relativamente boa, isso sugere que as misturas adicionadas não exercem grandes influências nos coeficientes gerados, o que sugere que nas misturas geradas há a possibilidade da mistura não estar influenciando nem concatenando no mesmo instante do áudio, algumas figuras foram plotadas mostrando que realmente não há grandes influências na mistura original e após isso foi tentado um método para encontrar a vogal na posição desejada baseado no tempo de áudio, para assim haver uma separação mais bruta.

Para gerar uma espécie de gabarito de quais são os instantes de tempo em que as vogais estão presentes, foi usada a similaridade de cossenos nos próprios coeficientes de MFCC, como na maior parte do áudio o silêncio está presente, como mostra a figura 5.5, ao transpor a matriz do MFCC, cada linha, nessa nova matriz, contém os coeficientes relativos a um intervalo de tempo, então quando fazemos a similaridade de cossenos na matriz transposta com ela mesma, temos uma matriz que terá uma separação bem clara, onde a maior região similar corresponderá ao silêncio e a outra região corresponderá à voz. Na figura 5.6 foi testado como o resultado se comportaria ao usar a métrica de distância de similaridade de cossenos e a distância euclidiana, como a distância euclidiana teve muitos valores despadronizados, como esperado, foi optado por fazer essa extração de instantes de tempo usando a similaridade de cossenos.

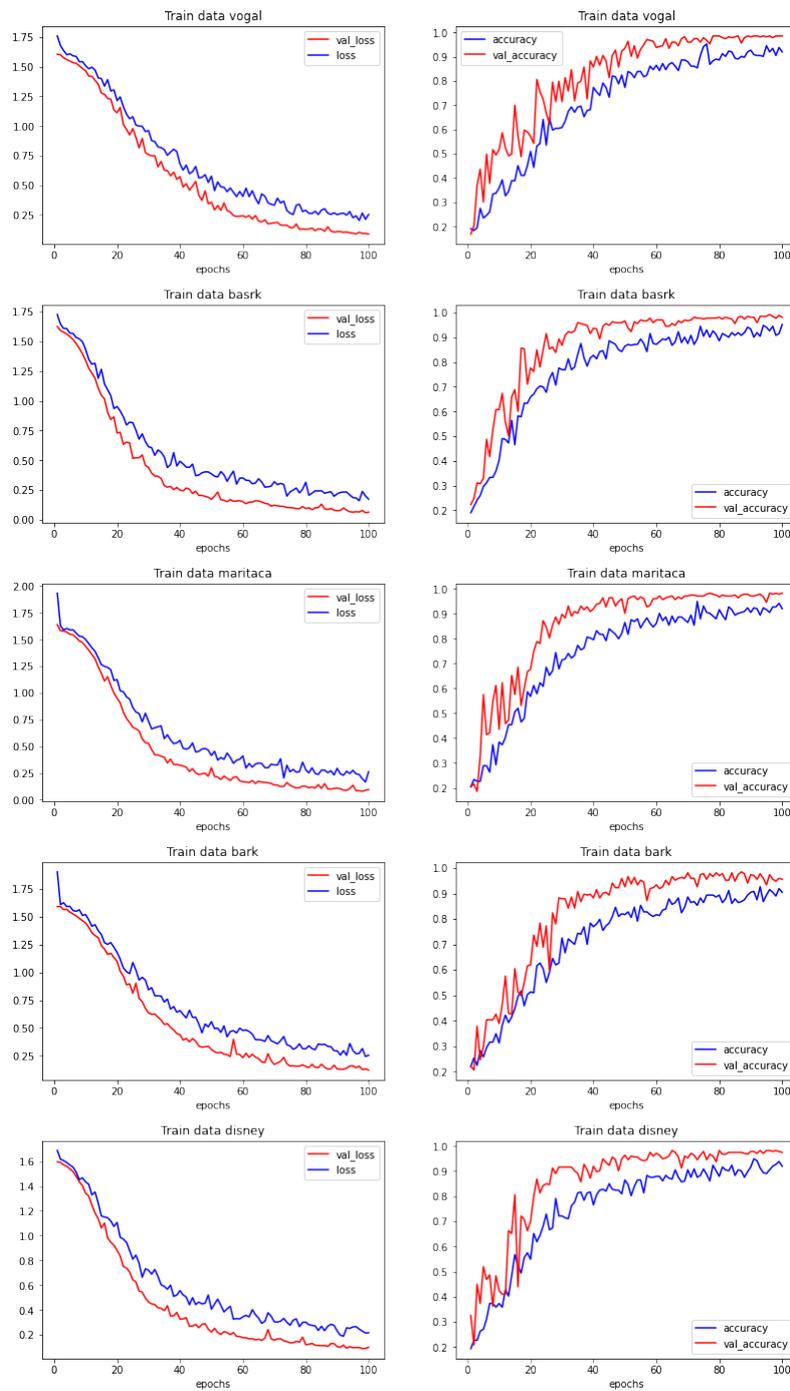


Figura 5.3: Os gráficos da esquerda representam o valor de perda em relação a quantidade de épocas, onde a curva vermelha mostra a perda relativa aos dados de treino e a azul aos de validação. Já os gráficos na direita mostram a acurácia da classificação por épocas, onde a linha azul representa os dados de treino e a linha vermelha a acurácia relativa aos dados de validação. As misturas colocadas foram vogal pura, chuva, som de maritacas, som de latidos, trecho de uma música instrumental, nessa ordem de cima para baixo.

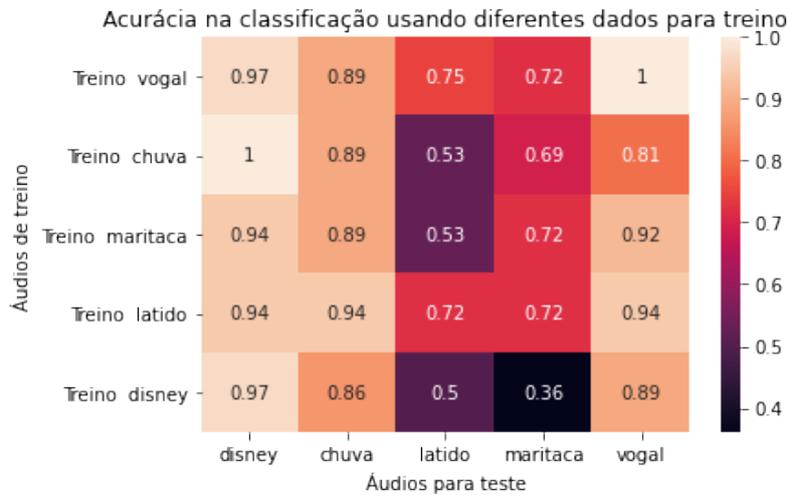


Figura 5.4: A figura representa a acurácia quando treinamos com os áudios misturados com os ruídos de chuva, uma música da Disney, latido, som de maritacas e a vogal pura e quando testamos com outros ruídos. Podemos ver que a maioria das classificações são relativamente boas.

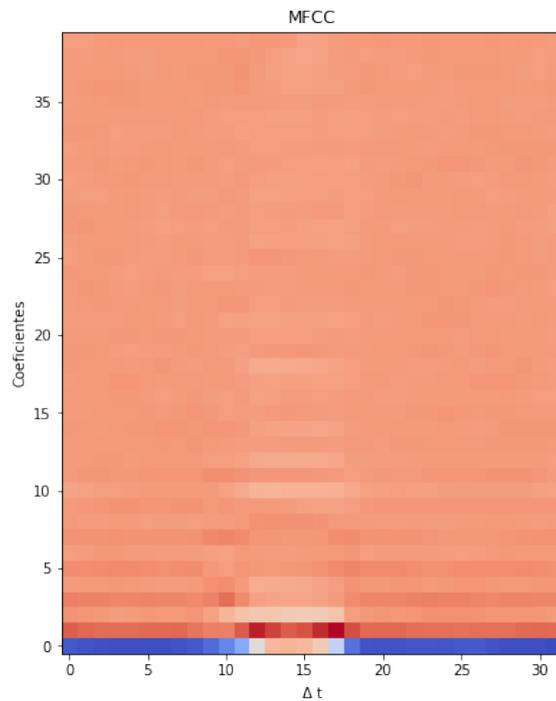


Figura 5.5: MFCC calculado de um áudio da vogal ‘a’ falada por um homem, a partir da figura podemos presumir que a vogal se encontra entre os instantes de $\Delta t = 10$ até $\Delta t \approx 18$.

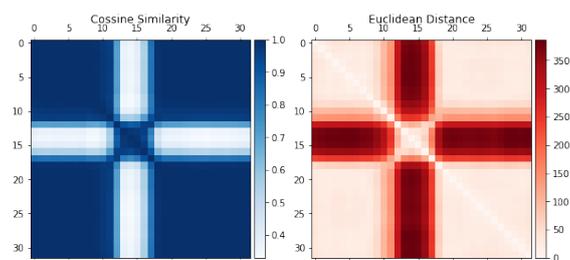


Figura 5.6: O gráfico da esquerda mostra o comportamento usando a similaridade de cossenos, já o da direita, a distância euclidiana, é possível observar que usando a distância euclidiana a variação numérica foi muito maior.

Para encontrar o tempo a partir do gráfico da similaridade de cossenos, o seguinte cálculo foi feito:

- Foi encontrado quanto tempo está contido em cada coluna da matriz do MFCC, para isso foi calculado o Δt de cada amostra, $\Delta t = \frac{1}{frequencia}$, como cada janela contém 512 amostras então cada coluna na matriz gerada pelo MFCC tem 512 amostras, logo, $\Delta t_{amostra} = 5128000$. Dado que existe esse $\Delta t_{amostra}$ agora só precisa extrair o menor intervalo de similaridade, pois como a vogal é minoria desse áudio, ela vai estar contida nesse intervalo. Se tomar como exemplo a figura gerada pela similaridade de cossenos que está presente na figura 5.6, o menor intervalo similar está bem no meio, como um quadrado no centro da figura, supondo que esse intervalo seja a 15ª coluna do MFCC à 25ª coluna, então o intervalo de tempo que a vogal está contida será entre $\Delta t_{15} = 0.96s$ e $\Delta t_{25} = 1.6s$.

Dado que existe a informação de onde a vogal está contida, poderá ser testado um método mais robusto a ruídos, que não se baseia na uniformidade do áudio fora do áudio da vogal. Para isso, o método será testado no mais simples dos casos, que é tentar recuperar o mesmo resultado de intervalo de tempo, usando o áudio só com vogais, sem nenhum ruído.

- A partir desse intervalo de tempo recuperado das vogais, o som das vogais foram extraídos para áudios puros, onde todo o conteúdo presente no áudio eram as vogais. Com esse novo banco de dados foi treinada uma nova rede neural, como os áudios não tinham o mesmo tamanho, em alguns deles foi necessário acrescentar algumas colunas de zeros ao final para deixá-los uniformes. Esse novo banco de dados foi separado em dados de treino e teste, na proporção de 70% para 30%
- Com essa nova rede neural treinada, foi aplicado o seguinte método: com os MFCC das vogais+silêncio, foram feitos os seguintes testes, se a coluna na extrema direita não influencia na classificação do áudio, ou seja, ela não melhora ou piora sua classificação, então muito provavelmente ela não contém nenhuma informação relevante, então essa coluna é substituída por uma coluna de zeros, passando assim para a próxima coluna, assim que o algoritmo encontrasse uma coluna que tivesse alguma informação relevante, ele começava o mesmo processo na outra extremidade, agora na extremidade esquerda, com isso, o algoritmo devolvia o intervalo que continha informações relevantes.

O método descrito apresenta algumas falhas e limitações, onde sua principal limitação é a impossibilidade de extrair o conteúdo da fonte desejada quando os

áudios são muito similares e superpostos. A tabela abaixo, mostra os resultados obtidos ao usar esse método. É possível observar que o método falha em encontrar os intervalos corretos na maioria dos casos, como esse método foi testado com todos os dados, então até mesmo nos áudios que são similares aos áudios que foram usados para o treinamento o método falhou. Como é possível ver na tabela, em 44.75% dos casos, o intervalo extraído não contém parte da vogal.

	$T_{c_0} < T_{o_0}$	$T_{c_0} = T_{o_0}$	$T_{c_0} > T_{c_0}$
$T_{c_f} < T_{o_f}$	$\sim 16.9\%$	$\sim 0.3\%$	0%
$T_{c_f} = T_{o_f}$	$\sim 19.68\%$	$\sim 1.02\%$	$\sim 1.78\%$
$T_{c_f} > T_{o_f}$	$\sim 31.44\%$	$\sim 2.81\%$	$\sim 26.07\%$

Tabela 5.2: Tabela que mostra o resultado geral dos tempos calculados onde a vogal começa e termina, T_{c_0} significa, Tempo calculado de início da vogal, T_{o_0} é o verdadeiro tempo de início da vogal, T_{c_f} é o tempo calculado em que a vogal termina, T_{o_f} é o tempo verdadeiro de término da vogal

5.2 Modelo Baseado na U-Net

Como não houveram resultados positivos no primeiro modelo, uma outra abordagem foi decidida. A segunda abordagem se baseia na similaridade do problema com o problema de segmentação de imagens, como já foi explicado nos capítulos anteriores, por isso a intenção do modelo é gerar uma máscara binária para cada áudio. Para lidar com esse desafio a rede neural escolhida foi a U-Net, com algumas modificações em relação à que foi apresentada nos capítulos anteriores.

A U-Net original é baseada numa entrada de 512x512, nesse experimento como entrada da rede foram usadas imagens geradas a partir dos espectrogramas. Nessas imagens a proporção da intensidade das frequências foi preservada e normalizada, para que o valor máximo presente seja 1. Quanto mais próximo um pixel de 1, maior a intensidade da frequência no espectrograma.

A figura 5.7 mostra a imagem do espectrograma do áudio com somente a vogal e ao seu lado sua máscara binária, apesar da figura do espectrograma da vogal parecer ser uma boa representação da vogal, usá-la como máscara binária é um pouco mais complicado, já que a intensidade dos pixels nessa imagem varia entre os valores de 0 e 1, por isso para gerar a máscara binária, o espectrograma do áudio foi gerado e após isso foi pego a intensidade máxima relativa a cada intervalo de frequência, ou seja, cada linha do espectrograma e com esses valores foi calculada a média dos máximos das frequências, onde caso a intensidade do intervalo de frequência num determinado intervalo de tempo é maior ou igual a esse número a máscara recebe um nessa posição, caso contrário 0, esse foi um método experimental que funcionou bem por causa de uma grande quantidade de silêncio na amostra.



Figura 5.7: A imagem do lado esquerdo representa a imagem salva do espectrograma e a imagem do lado direito a sua respectiva máscara binária

A métrica de acurácia usada foi a Interseção dentro da União (IoU score), essa métrica calcula a porcentagem da máscara prevista pela rede neural que se intercepta com a máscara desejada. Quanto mais próxima elas se parecem, mais perto de 1 é o resultado, figura 5.8

$$IoU = \frac{\text{Interseção}}{\text{União}} \quad (5.1)$$

Foram treinados alguns modelos, usando a proporção dos dados de 70% para treino, 21% validação e 9% teste.

Os primeiros testes foram realizados usando o mesmo tipo de ruído para todos os áudios, os gráficos abaixo ilustram os dois melhores resultados.

Nessa última configuração de rede foi testado a classificação para áudios com ruídos distintos:

	Loss	IoU score
Latido	1.6176	0.0922
Chuva	1.2296	0.1446
Instrumental	1.2303	0.1419
Maritaca	1.3613	0.1089

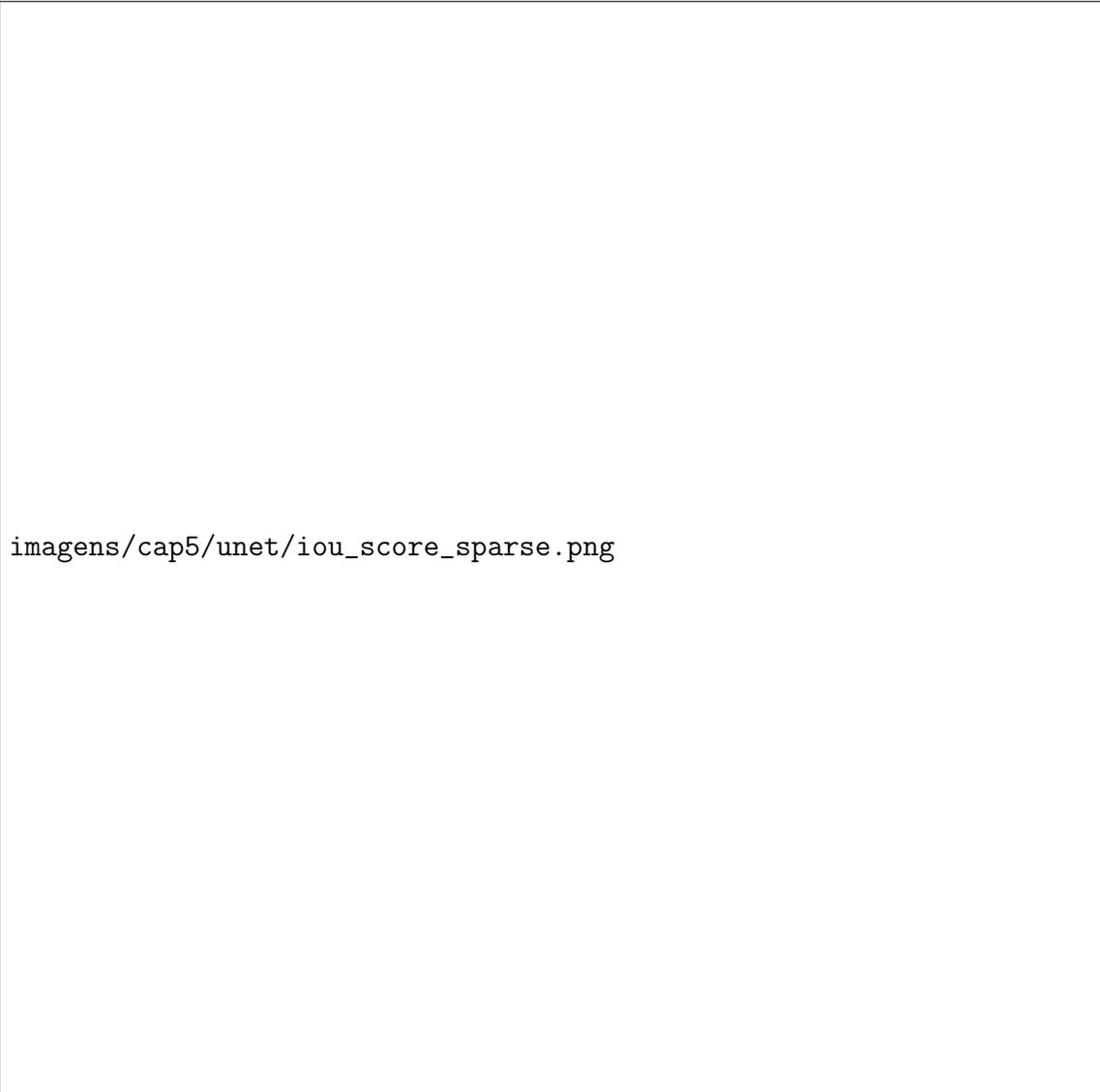
Tabela 5.3: Resultados obtidos usando as vogais puras para treino e dados com ruído para teste

	Loss	IoU score
sem ruído	0.6135	0.4632
ruídos diferentes	0.8867	0.2625
latido	1.0168	0.1757
chuva	1.0251	0.1743
instrumental	1.0306	0.1677
maritaca	1.0120	0.1840

Tabela 5.4: Resultados obtidos usando dados de treino e validação com diferentes tipos de ruídos diferentes

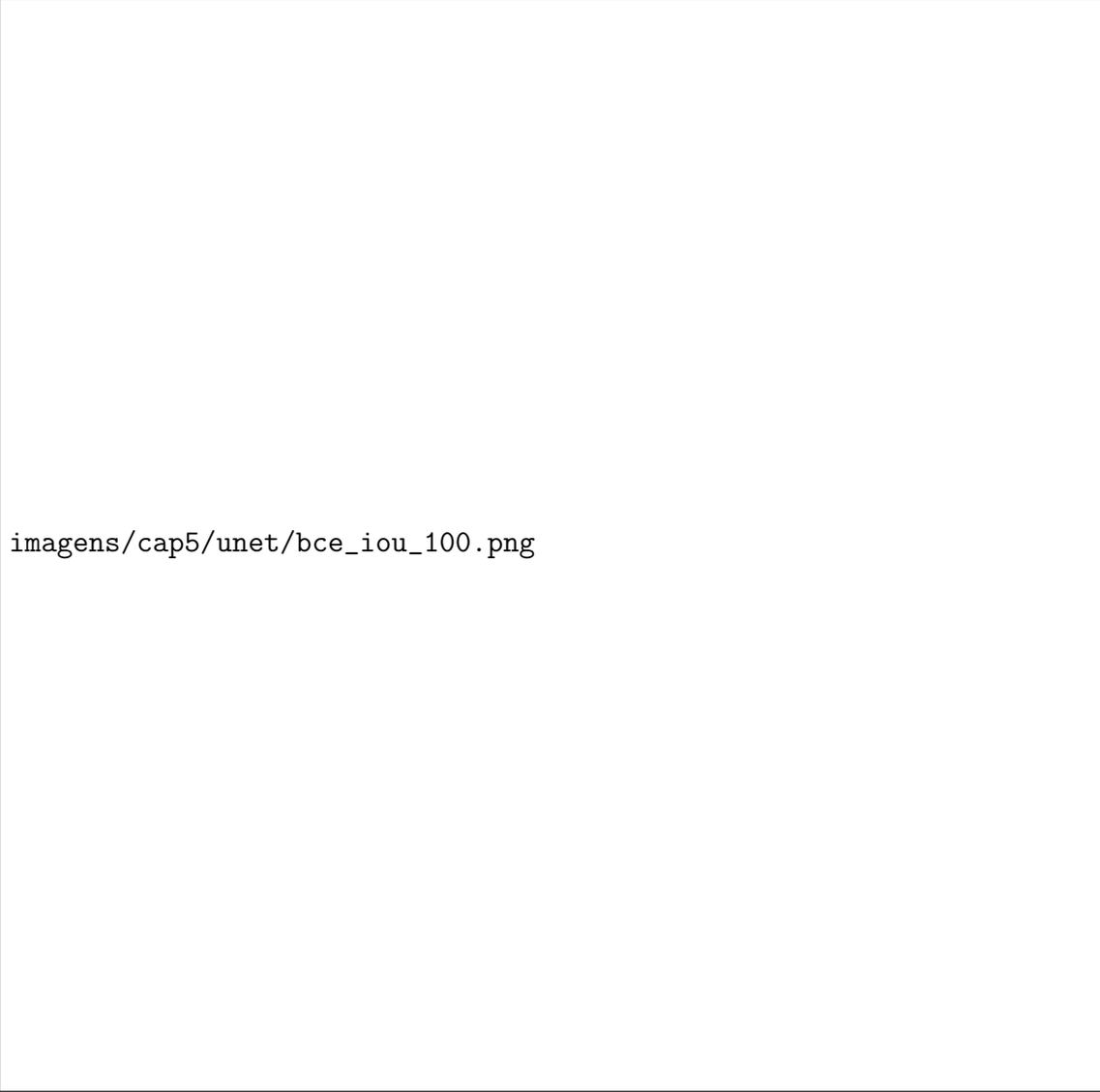
imagens/cap5/unet/iou.png

Figura 5.8: A imagem da esquerda mostra o caso em que não existe nenhum ponto interceptando as duas máscaras, a figura do meio quando parte das máscaras se interceptam e a da direita quando elas são exatamente iguais.



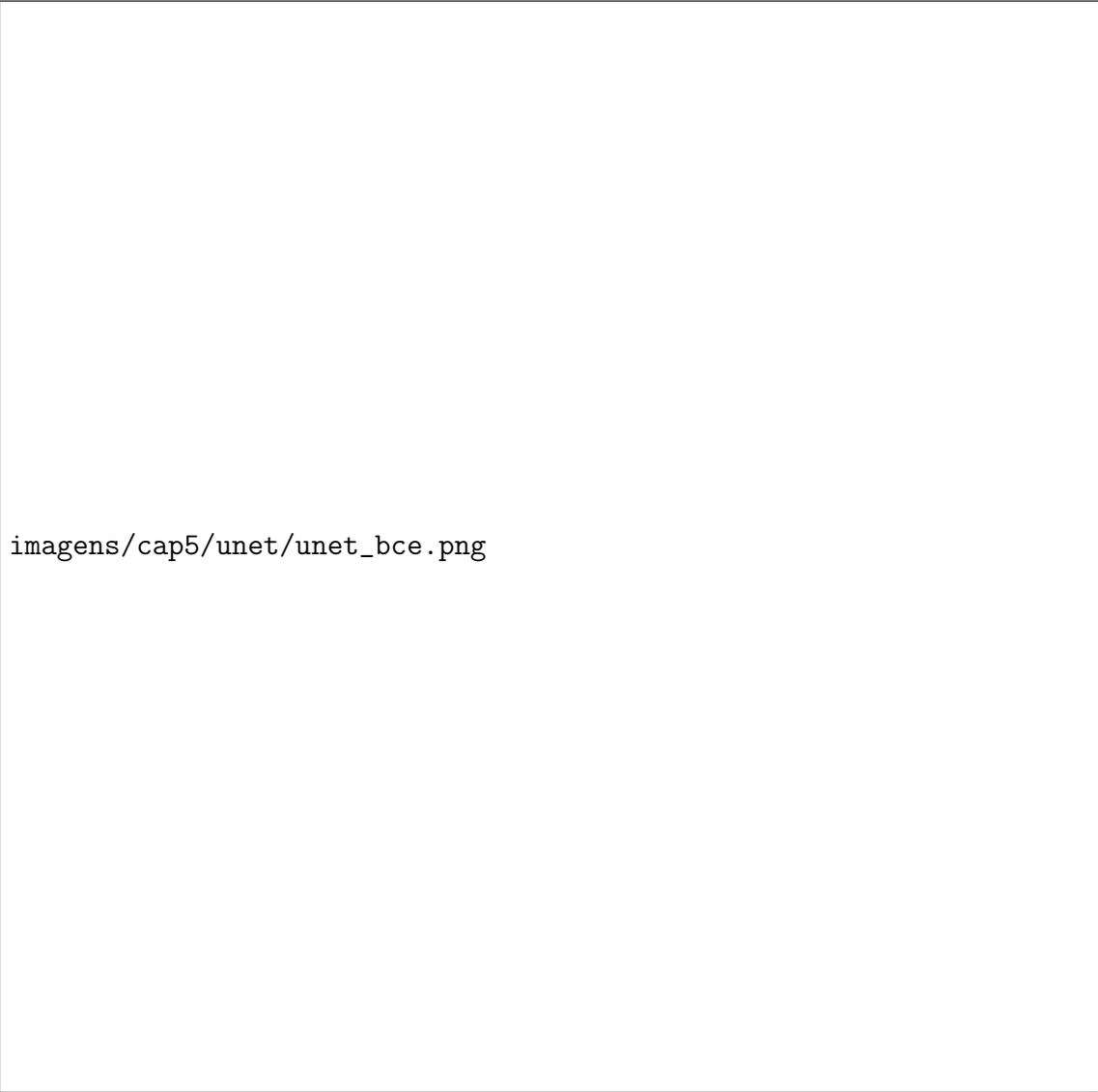
`imagens/cap5/unet/iou_score_sparse.png`

Figura 5.9: Otimizador: sgd, função de perda: sparse categorical cross entropy, na última camada a ativação é softmax, resultado nos dados de treino: perda: 0.032, e iou: 0.1241 .



`imagens/cap5/unet/bce_iou_100.png`

Figura 5.10: otimizador: Adam, função de perda: binary cross entropy, na última camada ativação: sigmoid, perda: 0.5865 - iou_score: 0.478, epochs : 100



imagens/cap5/unet/unet_bce.png

Figura 5.11: Curva de aprendizado da U-Net usando múltiplos ruídos juntos



`imagens/cap5/unet/img_mask_train_2.png`

(a) Exemplo de resultado



`imagens/cap5/unet/img_mask_train.png`

(b) Exemplo de resultado

Capítulo 6

Conclusões

Nessa dissertação foram testados dois modelos para a separação de sons que usaram métodos muito difundidos no campo de visão computacional, tal fato só é possível por causa da representação tempo-frequência de um áudio, uma maneira de mostrar o som como imagem, ao mudar o domínio tempo para TF e vice-versa não acarreta em mudanças significativas no conteúdo do sinal.

Os áudios que esse trabalho utiliza, são áudios muito simples e controlados para que possuam a menor quantidade de ruído e informação possível, por isso foi escolhido vogais. Os ruídos presentes nos áudios foram adicionados artificialmente após a coleta dos dados, gerando uma mistura controlada e artificial para gerar os experimentos.

O primeiro modelo proposto foi uma junção de CNN com MFCC, onde primeiro foi feita uma classificação e após uma tentativa de separação do sinal. Apesar da classificação dos sons do algoritmo utilizado ter tido uma eficácia boa, a separação dos sons mostrou-se insatisfatória, por isso foi necessário buscar outro modelo.

A segunda solução proposta foi trabalhar com a U-NeT juntamente com uma máscara binária. O conceito de máscara binária é bastante utilizado para segmentação de imagens, um campo muito explorado que já possui algoritmos mais recentes que a U-NeT. Quando a máscara binária é utilizada para os áudios, é necessário um threshold para criá-la, assemelhando-se muito à geração de máscaras binárias usando intensidade do pixel, método que é pouco utilizado na literatura mais recente.

A partir dessas IBM criadas e os espectrogramas dos áudios, foi possível gerar um banco de dados para o treinamento supervisionado para a rede. Os resultados do segundo método superaram os resultados do primeiro modelo sugerido, mas não se mostrou robusto a ruídos diferentes, apesar dos experimentos não terem mostrado um resultado satisfatório, modelos mais complexos que se utilizam do mesmo princípio de segmentação de imagens têm se tornado mais populares e oferecem um resultado bom[5].

Referências Bibliográficas

- [1] CHERRY, E. “Some experiments on the recognition of speech, with one and with two ears”, *Journal of the Acoustical Society of America*, v. 25, n. 5, pp. 975–979, 1953.
- [2] MCDERMOTT, J. H. “The cocktail party problem”, *Current Biology*, v. 19, n. 22, pp. R1024–R1027, 2009.
- [3] BREGMAN, A. “Auditory Scene Analysis: The Perceptual Organization of Sound”. v. 95, 01 1990. doi: 10.1121/1.408434.
- [4] BROWN, G. J., COOKE, M. “Computational auditory scene analysis”, *Computer Speech Language*, v. 8, n. 4, pp. 297–336, 1994.
- [5] HENNEQUIN, R., KHLIF, A., VOITURET, F., et al. “SPLEETER: A FAST AND STATE-OF-THE ART MUSIC SOURCE SEPARATION TOOL WITH PRE-TRAINED MODELS”. 2019.
- [6] KHAMPARIA, A., GUPTA, D., NGUYEN, N. G., et al. “Sound Classification Using Convolutional Neural Network and Tensor Deep Stacking Network”, *IEEE Access*, v. 7, pp. 7717–7727, 2019.
- [7] ARMITAGE, D. W., OBER, H. K. “A comparison of supervised learning techniques in the classification of bat echolocation calls”, *Ecological Informatics*, v. 5, n. 6, pp. 465–473, 2010.
- [8] LEE, Y.-C., CHI, T.-S., YANG, C.-H. “A 2.17-mW Acoustic DSP Processor With CNN-FFT Accelerators for Intelligent Hearing Assistive Devices”, *IEEE Journal of Solid-State Circuits*, v. 55, n. 8, pp. 2247–2258, 2020.
- [9] WANG, D., CHEN, J. “Supervised Speech Separation Based on Deep Learning: An Overview”, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, v. PP, 08 2017.
- [10] WANG, D. “Time–Frequency Masking for Speech Separation and Its Potential for Hearing Aid Design”, *Trends in amplification*, v. 12, pp. 332–53, 01 2009.

- [11] SOSE, S., MALI, S., MAHAJAN, S. “Sound Source Separation Using Neural Network”. In: *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1–5, 2019.
- [12] LUO, Y., MESGARANI, N. “Conv-TasNet: Surpassing Ideal Time-Frequency Magnitude Masking for Speech Separation”, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, v. PP, pp. 1–1, 05 2019.
- [13] MERTINS, A., MERTINS, D. “Signal Analysis: Wavelets, Filter Banks, Time-Frequency Transforms and Applications”, 11 2001.
- [14] ALLEN, J. “Applications of the short time Fourier transform to speech processing and spectral analysis”. In: *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, v. 7, pp. 1012–1015, 1982.
- [15] UNNISA, Y., TRAN, D., HUANG, F. “Statistical Independence, Measures and Testing”. 01 2014.
- [16] GUNAWAN, D., SEN, D. “Iterative Phase Estimation for the Synthesis of Separated Sources From Single-Channel Mixtures”, *Signal Processing Letters, IEEE*, v. 17, pp. 421 – 424, 06 2010.
- [17] WANG, Y., WANG, D. “Towards Scaling Up Classification-Based Speech Separation”, *IEEE Transactions on Audio, Speech, and Language Processing*, v. 21, n. 7, pp. 1381–1390, 2013.
- [18] OPPENHEIM, A., SCHAFER, R. “From frequency to quefrequency: a history of the cepstrum”, *IEEE Signal Processing Magazine*, v. 21, n. 5, pp. 95–106, 2004.
- [19] FAYEK, H. “Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What’s In-Between”. 2016. Disponível em: <<https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html>>.
- [20] GOODFELLOW, I., BENGIO, Y., COURVILLE, A. *Deep Learning*. MIT Press, 2016.
- [21] ACADEMY, D. S. “Deep Learning Book”. 2021. Disponível em: <<https://www.deeplearningbook.com.br/>>.

- [22] RONNEBERGER, O., FISCHER, P., BROX, T. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241, Cham, 2015.
- [23] DUMOULIN, V., VISIN, F. “A guide to convolution arithmetic for deep learning”, *ArXiv*, v. abs/1603.07285, 2016.