Pontifícia Universidade Católica
do Rio de Janeiro

Arthur Costa Serra

Reconstrução de músicas altamente degradadas usando modelos de aprendizado profundo

Dissertação de Mestrado

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre pelo Programa de Pós–graduação em Informática da PUC-Rio.

Orientador: Prof. Sérgio Colcher

Rio de Janeiro
Agosto de 2022

Pontifícia Universidade Católica
do Rio de Janeiro

Arthur Costa Serra

Reconstrução de músicas altamente degradadas usando modelos de aprendizado profundo

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre pelo Programa de Pós–graduação em Informática da PUC-Rio. Aprovada pela Comissão Examinadora abaixo.

Prof. Sérgio Colcher
Orientador
Departamento de Informática – PUC-Rio

Prof. Edward Hermann Haeusler
Departamento de Informática – PUC-Rio

Prof. Julio Cesar Duarte
Instituto Militar de Engenharia

Rio de Janeiro, 15 de Agosto de 2022

Arthur Costa Serra

Bacharel em Ciência da Computação pela Universidade Federal do Maranhão (UFMA) em 2020.

Ficha Catalográfica

Serra, Arthur Costa

Reconstrução de músicas altamente degradadas usando modelos de aprendizado profundo / Arthur Costa Serra; orientador: Sérgio Colcher. – Rio de janeiro: PUC-Rio, Departamento de Informática, 2022.

v., 54 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Inclui bibliografia

1. Audio Inpainting;. 2. Autoencoder;. 3. Reconstrução de música;. 4. Aprendizado profundo.. I. Colcher, Sérgio. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) por financiar parcialmente essa pesquisa sob o contrato 133263/2020-7.

Resumo

Serra, Arthur Costa; Colcher, Sérgio. **Reconstrução de músicas altamente degradadas usando modelos de aprendizado profundo**. Rio de Janeiro, 2022. 54p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A degradação da qualidade do áudio pode ter muitas causas. Para aplicações musicais, esta fragmentação pode levar a experiências altamente desagradáveis. Algoritmos de restauração podem ser empregados para reconstruir partes do áudio de forma semelhante à reconstrução da imagem, em uma abordagem chamada *Audio Inpainting*. Os métodos atuais de última geração para *Audio Inpainting* cobrem cenários limitados, com janelas de intervalo bem definidas e pouca variedade de gêneros musicais. Neste trabalho, propomos um método baseado em aprendizado profundo para *Audio Inpainting* acompanhado por um conjunto de dados com condições de fragmentação aleatórias que se aproximam de situações reais de deficiência. O conjunto de dados foi coletado utilizando faixas de diferentes gêneros musicais, o que proporciona uma boa variabilidade de sinal. Nosso melhor modelo melhorou a qualidade de todos os gêneros musicais, obtendo uma média de 13,1 dB de PSNR, embora tenha funcionado melhor para gêneros musicais nos quais os instrumentos acústicos são predominantes.

Palavras-chave

Audio Inpainting; Autoencoder; Reconstrução de música; Aprendizado profundo.

Abstract

Serra, Arthur Costa; Colcher, Sérgio (Advisor). **Quality enhancement of highly degraded music using deep learning-based prediction models**. Rio de Janeiro, 2022. 54p. Dissertação de mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Audio quality degradation can have many causes. For musical applications, this fragmentation may lead to highly unpleasant experiences. Restoration algorithms may be employed to reconstruct missing parts of the audio in a similar way as for image reconstruction — in an approach called audio inpainting. Current state-of-theart methods for audio inpainting cover limited scenarios, with well-defined gap windows and little variety of musical genres. In this work, we propose a Deep-Learning-based (DL-based) method for audio inpainting accompanied by a dataset with random fragmentation conditions that approximate real impairment situations. The dataset was collected using tracks from different music genres to provide a good signal variability. Our best model improved the quality of all musical genres, obtaining an average of 13.1 dB of PSNR, although it worked better for musical genres in which acoustic instruments are predominant.

Keywords

Audio Inpainting; Autoencoder; Music reconstuction; Deep Learning.

Sumário

Lista de figuras

Lista de tabelas

Lista de abreviaturas

VoIP – Voice over Internet Protocol

GAN – Generative Adversarial Networks

LSTM – Long-Short Term Memory

DCT – Discrete Cosine Transform

SiSEC – Signal Separation Evaluation Campaign

OMP – Orthogonal Matching Pursuit

SNR – Signal-to-Noise Ratio

TFNet – Time-Frequency Network

FMA – Free Music Archive

LPC – Linear Prediction Coding

STFT – Short-time Fourier Transform

ODG – Objective Difference Grades

BLSTM – Bidirectional Long-Short Term Memory

Hz – Hertz

dB – Decibels

MLP – Multilayer Perceptron

CNN – Convolutional Neural Network

BN – Batch Normalization

ReLU – Rectified Linear Units

GRC – Global Residual Connection

FPN – Feature Pyramid Network

RFB – Reverse Fusion Block

PSNR – Peak Signal-to-Noise Ratio

MSE – Mean Square Error

NRMSE – Normalized Root Mean Square Error

PEAQ – Perceptual Evaluation of Audio Quality

MOV – Model Output Variables

D_I – Distortion Index

1
Introdução

Todo arquivo digital de multimídia está sujeito a alguma forma de degradação do sinal. Essas degradações podem ocorrer pela avaria da mídia física, ruído no método de captura, falhas de *hardware* ou *software* durante a gravação ou, pelo meio de transmissão. Atualmente, o ambiente mais comum para a ocorrência de degradação de uma mídia é em aplicações de *Streaming* ou VoIP (*Voice over Internet Protocol*). Aplicações de tempo real como essas estão sujeitas a falhas de transmissão, problema conhecido como perda de pacotes. Como o sinal é transmitido em segmentos pela rede é possível que alguns deles se percam durante a transmissão, resultando em falhas na reprodução no destino. Desde o início da pandemia de 2019 houve um aumento considerável no número de usuário de aplicações de VoIP. Exemplo disso é o aplicativo Discord que em 2021 alcançou 150 milhões de usuários mensais (1).

Naturalmente, com o aumento do uso de ferramentas de transmissão ao vivo, os problemas do meio vão se tornando mais notáveis. A maioria dos usuários, algum momento, já passou por uma instabilidade de conexão que gera problema de perda de pacotes. Durante chamadas esse problema pode ser percebido pela "robotização" da voz. Dependendo da largura de banda cada pacote pode representar de 2,5 até 64 milissegundos de áudio, em média são utilizados 4 milissegundos para cada pacote (2). A partir de 10% de perda o efeito de robotização já começa a ser percebido, quanto maior a taxa de perda, maior o efeito de degradação. Na maioria dos protocolos, esse problema é tratado simplesmente completando os segmentos perdidos do sinal com silêncio ou repetindo os segmentos íntegros recebidos (3).

Considerando mídias sensíveis como músicas, onde qualquer alteração do sinal afeta diretamente a experiência, surge a necessidade de tratar essa falha de modo mais inteligente. Atentando o tratamento de adição de silêncio, o problema assemelha-se a um campo de estudo já explorado, chamado *Image Inpainting*. Esse problema consiste em recuperar segmentos perdidos a partir de um contexto íntegro. Assim, em 2011 surge o primeiro trabalho de *Audio Inpainting* enquanto tarefa análoga a *Image Inpainting* (4).

Durante muitos anos o problema ficou em hiato, até que começou a ser novamente notado em 2019, com métodos mais modernos de restauração

de áudio baseados em aprendizado profundo. Alguns destes trabalhos são focados em melhorar a qualidade de sinais fala (5, 6, 7) e outros sinais de música (8, 9, 10). Contudo, a maioria dos trabalhos publicados abrangem apenas cenários limitados de reconstrução. Alguns, em fase inicial cobrem apenas cenários de degradação bem simples, com conjuntos de dados pequenos, pouca variedade de sinal e degradações leves com posições e tamanho de lacunas conhecidos. A maioria considera apenas cortes únicos e centralmente posicionados em faixas integras e apenas o gênero instrumental compõe os experimentos. Assim, por não haver um padrão de avaliação e nem um conjunto de dados relevante, as pesquisas no campo ocorrem de forma mais lenta que no campo de visão computacional, por exemplo.

Em vista disso, neste trabalho temos dois focos principais. Construir um conjunto de dados musicais robusto, com 13.583 faixas e 16 gêneros, bem como aplicar degradações de quantidades e posições aleatórias que mais se aproximem de situações reais. Além de definir uma metodologia de reconstrução que possa ser avaliada de forma padronizada independente do método de restauração escolhido. Avaliando tanto etapas intermediarias de restauração quanto o real impacto perceptível final. Em complemento, explorar o real potencial de algoritmos de aprendizado profundo *Autoencoders* na resolução da tarefa, treinando diversos modelos de modo a definir um *baseline* comparável para o conjunto proposto. Para isso realizamos treinamento em camadas de 11 modelos diferentes para selecionar o melhor modelo segundo as métricas de reconstrução, e, por conseguinte, avaliá-lo através da métrica de percepção objetiva de modo entender as relações dos sinais.

1.1 Organização do trabalho

Esta proposta de dissertação está estruturada da seguinte forma. No Capítulo 3, descrevemos os trabalhos mais relevantes relacionados a este. No Capítulo 2, fazemos uma revisão da fundamentação teórica necessária para compreensão deste trabalho, como algoritmos de representação de áudio e redes neurais utilizadas. Seguido pelo Capítulo 4, onde definimos a metodologia de reconstrução de áudio proposta para resolução do problema de *Audio Inpainting*. No Capítulo 5, apresentamos a construção do conjunto de dados proposto, as métricas de avaliação utilizadas, os parâmetros e esquema de treinamento e os resultados finais obtidos do processo. Por fim, no Capítulo 6, apresentamos as considerações finais do trabalho, publicações geradas e trabalhos futuros previstos para complementar a pesquisa.

2
Fundamentação Teórica

Neste capítulo são apresentados os principais fundamentos utilizados no desenvolvimento da metodologia proposta para resolução do problema de reconstrução de áudio. Descrevendo os algoritmos utilizados para extração de características de áudios, bem como os *Autoencoders* utilizados neste trabalho.

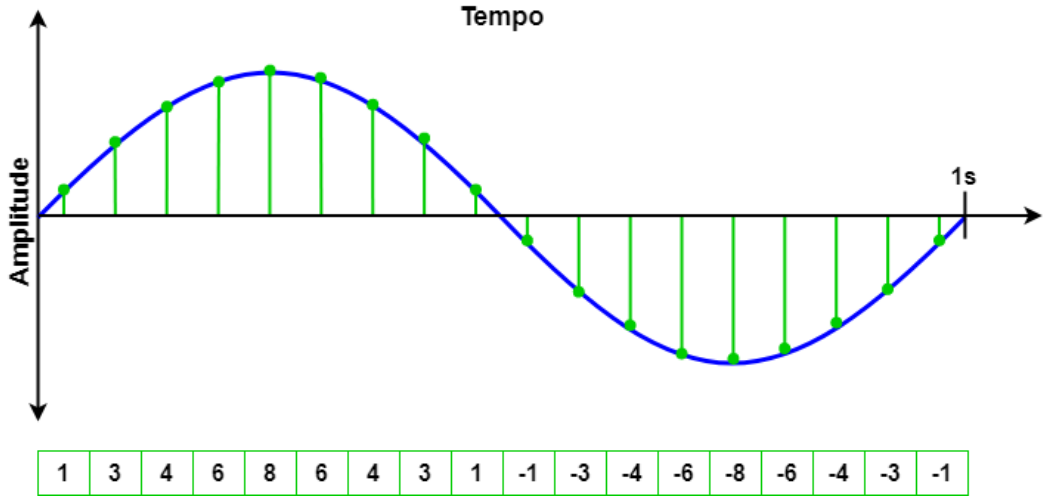
2.1
Representações de áudio

Antes de visitarmos algoritmos de representação de áudio precisamos entender o que é e como interpretamos um sinal sonoro. Um som é produzido a partir da variação da pressão atmosférica através de vibração, produzindo um sinal oscilante entre alta e baixa pressão. Assim, computacionalmente, esse sinal é capturado como uma sequência temporal de valores inteiros, que corresponde tal variação em forma de onda como "áudio". Esta sequência pode ser uma representação densa ou esparsa, quanto mais denso, maior a qualidade do áudio, também descrita pelo termo "*sample rate*", ou taxa de amostragem. Equivalente à unidade de frequência do sistema internacional *Hertz* (Hz), a taxa de amostragem define a quantidade de unidades de representação a cada segundo de áudio. Além disso, cada unidade possui uma profundidade de precisão definida por bits, a quantidade de bits também define a fidelidade de representação dos sons, geralmente variando de 8 a 256 bits.

Como ilustra a Figura 2.1, para um hipotético sinal sonoro representado pela curva azul, sua contrapartida em áudio é um vetor com os valores pontualmente selecionados na curva. Neste exemplo, temos a representação de um áudio com taxa de amostragem de 18Hz e amplitude de 4 bits, tal amplitude também pode ser interpretada como um intervalo com 2^4 valores inteiros que variam de -8 a 8.

Contudo, no mundo real a taxa de amostragem e a profundidade de bits são bem maiores. Visto que a audição humana consegue interpretar sons na faixa de 20-20.000Hz (11), sem considerar a profundidade de bits, é necessário um vetor de 20.000 posições para representar um segundo de áudio. Além disso, o teorema de amostragem Nyquist-Shannon diz que para não haver erro significativo na representação, a taxa mínima de amostragem deve ser pelo

Figura 2.1: Demonstração da representação de um sinal para áudio.

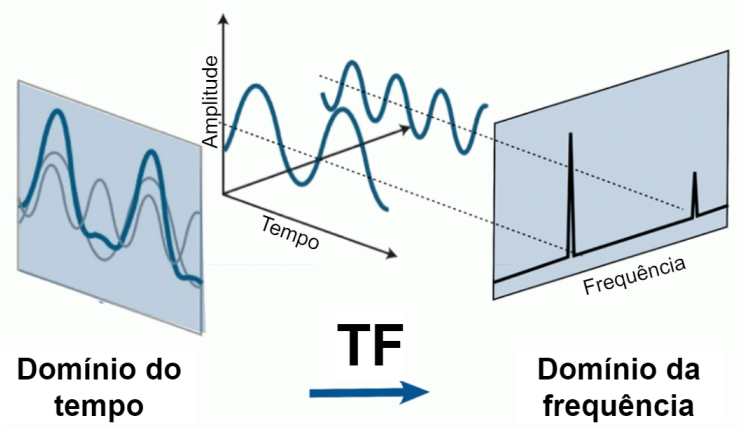


Fonte: Autor

menos duas vezes maior que o limiar desejado (12). Sendo assim, considerando que o ouvido humano capture uma frequência de até 20kHz, uma representação fiel deve conter no mínimo 40kHz de taxa de amostragem. Áudios de um CD contêm 44kHz de representação com 16 bits de profundidade, portanto podemos perceber a ordem de grandeza que possuem os vetores de áudios.

Para as tarefas que envolvem processamento de sinal, interpretar dados de áudio diretamente no domínio do tempo é uma tarefa de alto custo computacional. Por isso, a solução mais comum para esse problema é trazer a representação para o domínio das frequências, geralmente utilizando a Transformada de Fourier, representado na Figura 2.2.

Figura 2.2: Transformada de fourier.



Fonte: Adaptado de Range et al. (13)

A Transformada de Fourier é o método matemático para expressar qualquer sinal em um somatório de funções seno e cosseno (14), sendo o cosseno a representação da amplitude ($\mathbb{R}$) e o seno a representação de fase ($\mathbb{C}$)

da frequência complexa resultante. Formalizado pela Equação 2-1, para uma dada função $f(x)$ suas frequências ξ são descritas através da amplitude e fase complexa pela função $\widehat{f}(\xi)$.

$$\widehat{f}(\xi) = \int f(x)(cos(2\pi\xi x) - isen(2\pi\xi x))dx \tag{2-1}$$

Contudo, se quisermos trazer um sinal de volta a forma original após uma transformada, o Teorema da Inversão de Fourier diz que se aplicarmos o mesmo processo em um sinal transformado obteremos o sinal original (15).

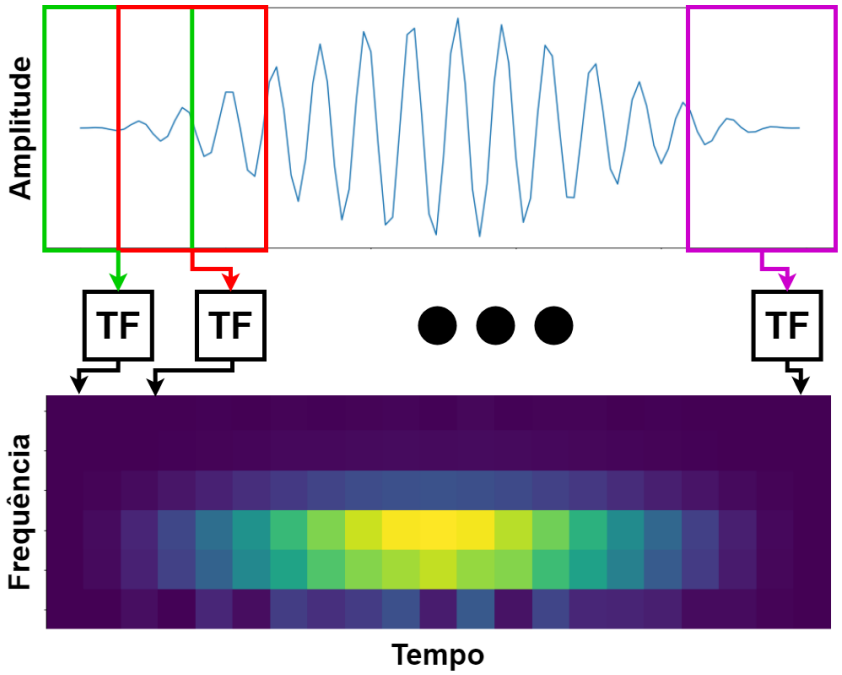
$$f(x) \leftrightarrow \widehat{f}(\xi) \tag{2-2}$$

Nas seguintes subseções serão descritos dois algoritmos populares de representações de áudio derivados da transformada de Fourier e utilizados neste trabalho para interpretar sinais densos.

2.1.1 STFT - *Short-time Fourier Transform*

O algoritmo STFT transforma um sinal do domínio do tempo para o domínio tempo-frequência. Como ilustrado na Figura 2.3, o processo ocorre por uma sequência de transformadas de fourier, em pequenas janelas deslizantes, através do sinal em forma de onda. Assim, gerando informações da variação de frequência temporalmente, diferente da transformada de fourier padrão que traz apenas uma informação média geral do sinal de entrada (16).

Figura 2.3: Processo de execução do STFT.



Fonte: Autor

Para a execução desse algoritmo são necessárias algumas informações de entrada além do sinal, como a largura da janela e o tamanho do salto no janelamento, visto que o tamanho da janela não pode ser maior que a metade do comprimento total do sinal. Também, para manter a sobreposição das janelas e preservar a fidelidade de representação, o tamanho do salto não pode ser maior que a largura da própria janela,

Geralmente definimos o tamanho do salto como $\frac{1}{2}$, $\frac{1}{4}$ ou $\frac{1}{8}$ do tamanho da janela. Vale atentar para um balanceamento na largura da janela, pois quanto menor a largura da janela maior a resolução temporal e menor a resolução das frequências e vice-versa. Utilizando os mesmos parâmetros de transformação, o algoritmo é inversível através do janelamento de transformadas inversas.

Por fim, para conseguirmos uma visualização em forma de espectrograma, como na Figura 2.3, o componente complexo das transformadas é descartado e a componente real é convertida apenas para valores absolutos para visualização, obtendo um espectrograma de magnitudes.

Desta feita, percebemos que o princípio da inversão não funciona em um espectrograma incompleto, pois não possui mais a componente de fase.

2.1.2 Griffin-Lim

O algoritmo de Griffin-Lim é um método de reconstrução de fase baseado na redundância do STFT (17). Essa redundância se da pela consistência do espectro complexo entre as sobreposições do janelamento. Deste modo, através de algumas iterações, sem nenhum conhecimento prévio do sinal original, o algoritmo consegue projetar a fase complexa de um espectro (18).

O algoritmo segue a Equação 2-3, onde S representa o espectrograma complexo que será reconstruído, P_x uma projeção em um conjunto x, m o índice de iteração, C um conjunto de espectrograma consistente e A um conjunto de espectrogramas cuja amplitude é mesma do sinal original.

$$S^{n+1} = P_C(P_A(S^n)) \tag{2-3}$$

Ambas as projeções podem ser melhor expressas pelas Equações 2-4 e 2-5, onde φ e $\varphi^\dagger$ representam, respectivamente, o resultado do STFT e seu pseudo inverso para um espectrograma complexo S. A projeção P_A é basicamente uma iteração sobre uma multiplicação entre a amplitude inicial A com o espectrograma S, dividido pela amplitude de S.

$$P_C(S) = \varphi\varphi^\dagger S \tag{2-4}$$

$$P_A(S) = A \odot S \oslash |S| \tag{2-5}$$

Em suma, o algoritmo Griffin-Lim pode ser dito como um processo de minimização de erro, como descrito na Equação 2-6, onde $||\ ||_{Fro}$ é a norma de Frobenius (19).

$$\min_{S} ||S - P_C(S)||^2_{Fro},\ onde\ S \in A \tag{2-6}$$

Também pode ser descrito como:

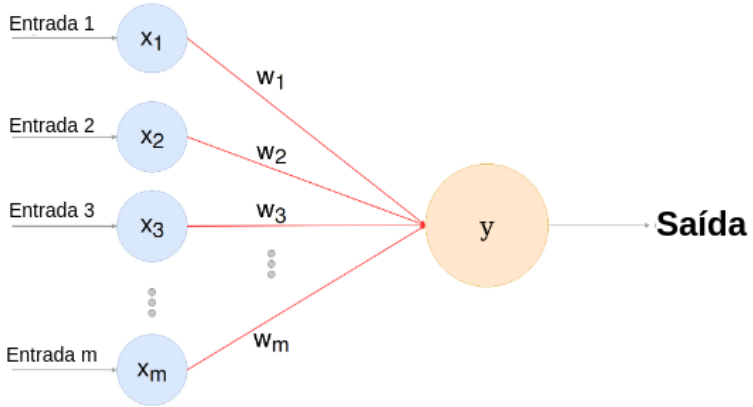
Algorithm 1 Griffin-Lim

$S \leftarrow Espectrograma\ \mathbb{R}$
$A \leftarrow S + Identidade\ \mathbb{C}$
$y \leftarrow \varphi^{\dagger}(S)$
while $i < n$ **do**
  $S' \leftarrow \varphi(y)$
  $S' \leftarrow A \odot S' \oslash |S'|$
  $y \leftarrow \varphi^{\dagger}(S')$ ▷ y sinal final gerador de S
end while

2.2 Redes Neurais Artificiais

Rede Neural Artificial é um modelo matemático projetado para simular uma rede neural biológica. Um neurônio, do ponto de vista biológico, é um tipo de célula presente no sistema nervoso que serve para transmitir informações através de impulsos elétricos. Um neurônio artificial, originalmente chamado de *perceptron*, exerce a mesma função de propagação de informação em uma cadeia de transmissão, análoga ao sistema nervoso, possibilitando a construção de diversos modelos de inteligência artificial. Proposto por Rosenblatt et al. (20), um *perceptron* é composto por uma camada de entrada, uma camada de pesos e uma camada de saída como ilustra a Figura 2.4.

Figura 2.4: Representação de um perceptron.



Fonte: Adaptado de Arc et al. (21)

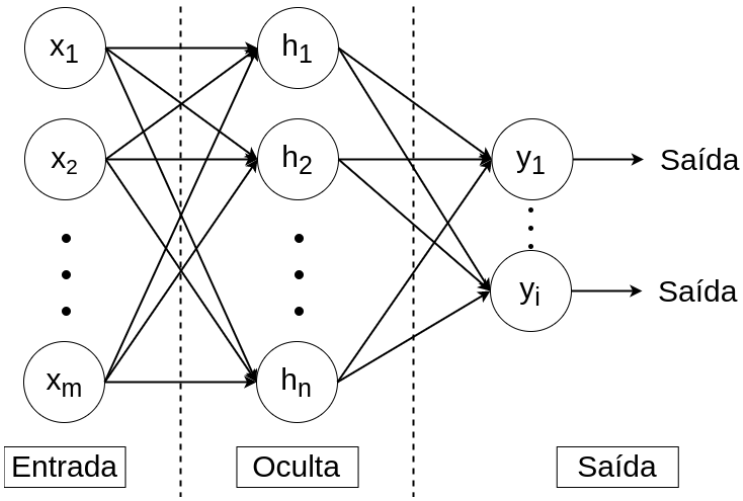
Tendo em vista x_m como entradas de valores reais, onde w_m são os pesos que ponderam as respectivas entradas. Por fim, na camada de saída temos a

função y, definida pela Equação 2-7, que define um valor chamado *score* que representa a entrada do perceptron. O ajuste dos pesos w_m define o processo de aprendizado de um neurônio. Contudo, estruturas de neurônios simples como esta só resolvem problemas lineares. Para atenuar esse problema é introduzido na camada de saída uma função de ativação, que basicamente decide se o neurônio deve ou não ser considerado.

$$y = \sum_{i=0}^{m} x_i w_i \tag{2-7}$$

A rede neural mais simples possível é chamada MLP (*Multilayer Perceptron*). Essa rede também é composta de camadas, além do conjunto de neurônios de entrada e saída é adicionado um conjunto intermediário chamado camada oculta, como demonstra a Figura 2.5.

Figura 2.5: *Multilayer perceptron*



Fonte: Autor

Em vista disso, resta saber como funciona o processo de aprendizado dessas redes. Inicialmente, todos os pesos são definidos aleatoriamente a partir da camada de entrada até a camada de saída. Posteriormente, é calculado o erro associado a cada peso através de derivada da função de custo, chamada de gradiente. O gradiente indica a direção de ajuste para redução do erro. Para atualizar os pesos, uma taxa de aprendizado (*learning rate*) deve ser definida para determinar os novos valores dos pesos a cada iteração como definido na Equação 2-8, onde lr define a taxa de aprendizado e δ a derivada do custo associado (22). Assim, atualizando os pesos das camadas de saída até a camada de entrada define-se o processo de aprendizado *Backpropagation*.

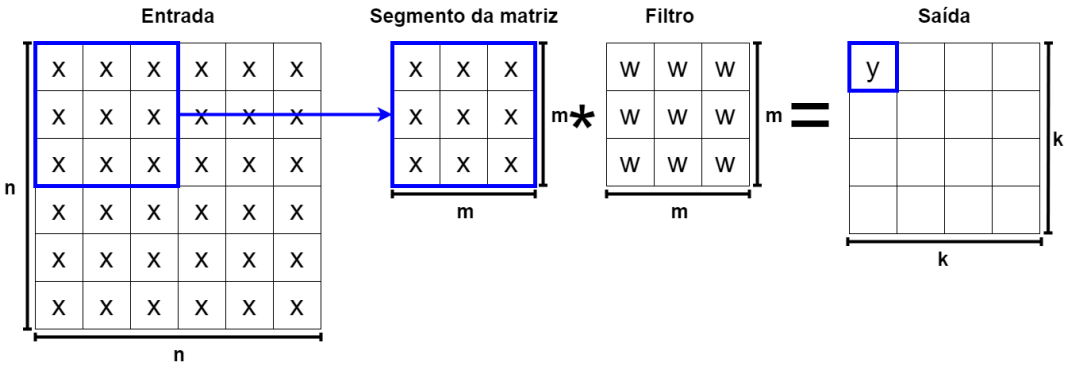
$$peso = peso - (lr * \delta) \tag{2-8}$$

2.3 Rede Neural Convolucional

Uma rede neural convolucional, ou simplesmente CNN (*Convolutional Neural Network*), é um tipo de rede neural pensada para solucionar problema de visão computacional, ou seja, uma maneira de interpretar dados bidimensionais de modo a considerar a composição espacial. Essas redes normalmente possuem 3 camadas principais, uma convolucional, uma de *pooling* e uma totalmente conectada.

A Figura 2.6 ilustra o funcionamento da operação de convolução. Para toda matriz de entrada com dimensão $[n \times n]$ com valores $x \in \mathbb{R}$, é definido um filtro de dimensão $[m \times m]$ com valores $w \in \mathbb{R}$, que multiplica um segmento da entrada de tamanho equivalente e gera um valor y na matriz de saída de dimensão $[k \times k]$. Sabendo que, $y = \sum_{i=0}^{m} \sum_{j=0}^{m} (x_{(i,j)} * w_{(i,j)})$ e $k = (\frac{n-m}{p} + 1)$, sendo p o tamanho do passo do deslizamento do segmento na matriz de entrada.

Figura 2.6: Camada convolucional.

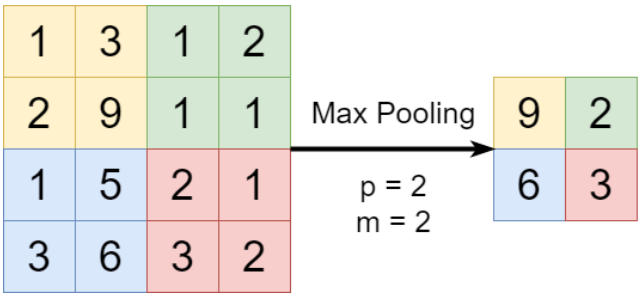


Fonte: Adaptado de Reynolds et al. (23).

Já a camada de *pooling* consiste em uma operação de redução de dimensionalidade, a fim reduzir a complexidade de operações com a matriz de saída da convolução, evitar o sobre-ajuste de aprendizado e extrair características mais representativas. Esta operação pode ser realizada de duas formas, máxima ou média. O *pooling* máximo, através de uma operação de convolução, retorna o valor máximo de um seguimento de dimensão $[m \times m]$ para uma matriz resultante, como demonstrado na Figura 2.7. A operação de *pooling* médio efetua o mesmo processo, porém com o valor médio do seguimento.

Por fim, a camada totalmente conectada é uma rede neural com estrutura e funcionamento análogos ao de um MLP, que recebe como entrada cada posição da matriz resultante da camada *pooling*. Uma CNN profunda consiste em uma sequência de camadas convolucionais e de *pooling* e, dependendo do objetivo da rede, uma camada totalmente conectada.

Figura 2.7: Camada de *pooling* máximo.



Fonte: Adaptado de Reynolds et al. (23).

2.4 Autoencoder

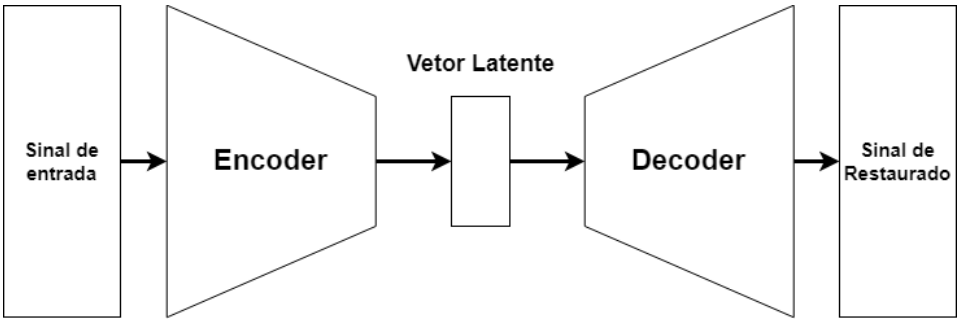
Autoencoder é um tipo de rede neural que tenta imitar da melhor maneira possível uma determinada entrada. Para isso, a rede transforma a entrada em um vetor de representação, chamado vetor latente, e a partir dessa representação reconstrói a entrada apenas com o comportamento aprendido (24). Formalmente, podemos dizer que um *Autoencoder* produz uma relação não linear entre uma entrada e uma saída através de uma representação latente.

Como ilustra a Figura 2.8, a primeira etapa (*Encoder*) tem a função de reduzir a dimensionalidade da entrada com o mínimo de perda possível, enquanto o segunda etapa (*Decoder*) pode exercer diversos papéis como segmentação, redução de ruído e reconstrução de falhas. Desta feita, vale ressaltar algumas observações sobre *Autoencoders*:

- Compressão específica: apenas os dados parecidos com o contexto de treinamento serão comprimidos corretamente de modo a restaurá-los com qualidade.
- Aprendizado sem supervisão: os dados de treinamento não precisam ser rotulados, qualquer dado de entrada é um contexto valido.
- Reconstrução com perdas: qualquer sinal reconstruído a partir do vetor latente está sujeito e perda de fidelidade no processo de compressão.

A partir dessas observações, podemos considerar que, para usá-la corretamente na tarefa de *Audio Inpainting*, precisamos: (a) um contexto bem definido de dados e como boa diversidade de representação; (b) o áudio precisa estar em uma representação de duas dimensões, de modo a construir um contexto espacial do sinal; (c) considerar as falhas da representação de saída para avaliar o desempenho do modelo. Desta forma, espera-se que o *Encoder* consiga construir uma representação codificada (vetor latente) que represente o comportamento do sinal de tal forma que o *Decoder* consiga suprimir as falhas.

Figura 2.8: Representação de um autoencoder.

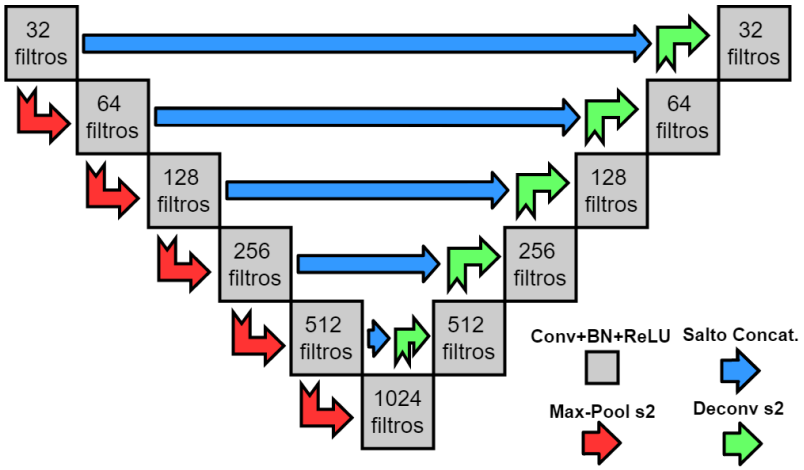


Fonte: Autor

2.4.1 U-Net

Criada com objetivo de segmentação de imagens médicas, a U-Net (25) é o *Autoencoder* mais conhecido da literatura. Sua arquitetura consiste em um caminho de contração, para capturar o contexto e construir um vetor latente, e um caminho simétrico de expansão que, a princípio, foi pensado para segmentar regiões de interesse. Na arquitetura original existem algumas restrições de entrada, por exemplo, a imagem de entrada precisa ter altura e largura iguais e precisa ter um tamanho que é uma potência de dois.

Figura 2.9: Arquitetura U-Net.

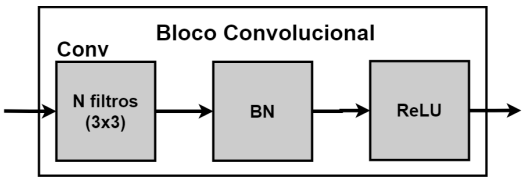


Fonte: Autor

Na arquitetura U-Net utilizada neste trabalho, ilustrada na Figura 2.9, o processo de *encoder* incluindo o vetor latente é composto de 6 blocos convolucionais. Um bloco convolucional, representado na Figura 2.10, consiste na sequência das operações convolução de n filtros $[3 \times 3]$, *Batch Normalization* (BN) e ativação ReLU (*Rectified Linear Units*). A BN é uma operação que reescala valores de tal modo que produza de um espaço de parâmetros e

gradientes mais suaves (26). Já a operação ReLU é uma ativação básica que determina zero para entradas negativas e a própria entrada, caso contrário.

Figura 2.10: Estrutura de um bloco convolucional.



Fonte: Autor

A saída de cada bloco convolucional tem dois destinos distintos, um para uma operação de *pooling* máximo que reduz as dimensões de altura e largura pela metade e direciona para a entrada do próximo bloco, e o outro caminho é um salto de concatenação com a camada simétrica no processo de *decoder*. A cada bloco de contração dobra-se a quantidade de filtros.

O processo de *decoder* consiste em 5 blocos convolucionais, que recebem como entrada uma convolução transposta (Deconv) concatenada à camada de contração com dimensão equivalente. A Deconv é uma convolução com passos iguais a 2 que dobra a dimensão de saída e reduz a quantidade de filtros pela metade.

2.4.2 U-Net V2

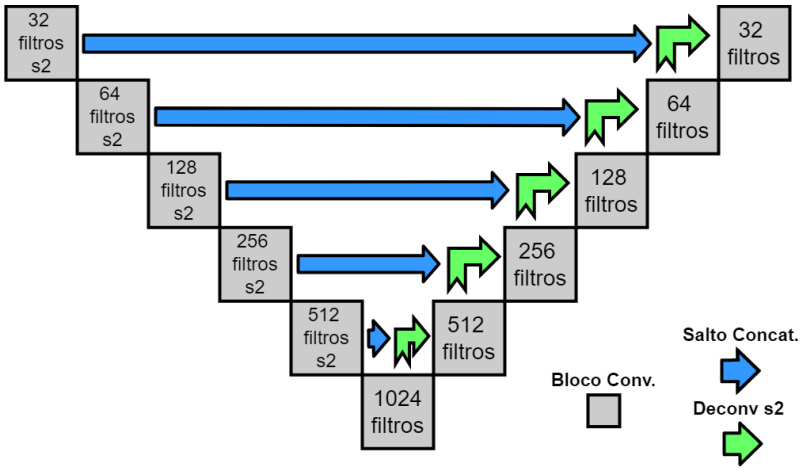
Esta arquitetura é uma varição do modelo U-Net que não usa operações de *pooling* para o processo de compressão da entrada. Em alguns contextos, os valores adjacentes de uma matriz tem uma lógica maior que apenas a representação visual, nesses contextos uma operação que preserva apenas a informação máxima de uma região perde mais informação que o necessário.

Em vista disso, a solução para reduzir a dimensão de uma entrada sem alterar a lógica posicional dos valores é utilizar convoluções espaçadas. Ilustrada na Figura 2.11, as operações de *pooling* são abolidas e a convolução dos blocos de compressão tem filtro $[3 \times 3]$ e com passos de deslizamento de 2 unidades.

2.4.3 Res-U-Net

Ilustrada na Figura 2.12, a arquitetura Res-U-Net (27) é uma arquitetura U-Net que utiliza um mecanismo chamado GRC (*Global Residual Connection*). Esse mecanismo tem se mostrado tendência em modelos de aprendizado profundo em tarefas como *denoising* e super-resolução (28, 29). Basicamente, a saída da U-Net passa por uma convolução com um filtro $[3 \times 3]$ e o resultado

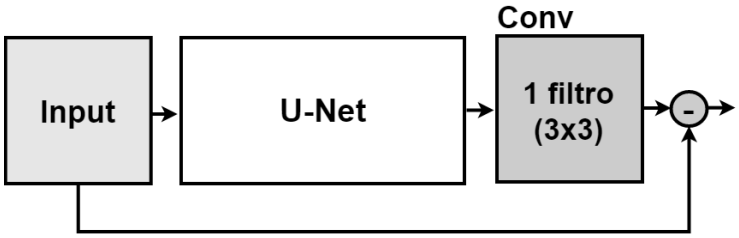
Figura 2.11: Arquitetura U-Net V2.



Fonte: Autor

dessa convolução passa por uma operação de subtração com a entrada original da U-Net para produzir uma reconstrução com características residuais.

Figura 2.12: Arquitetura Res-U-Net.

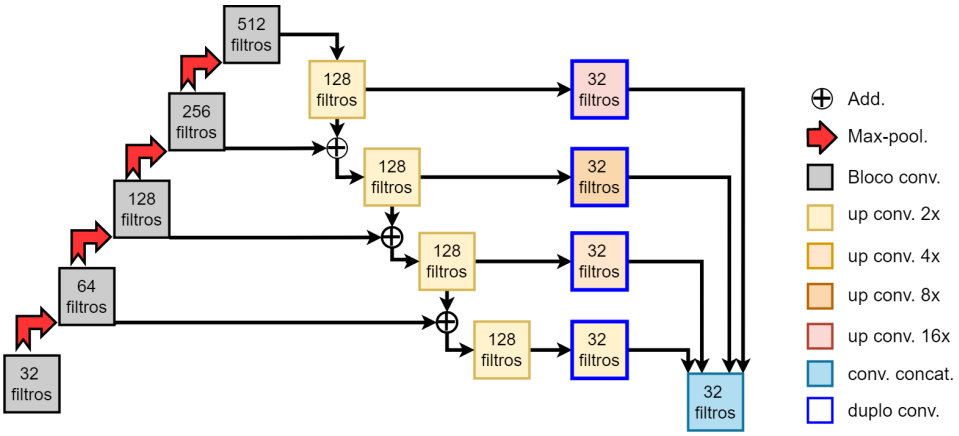


Fonte: Autor

2.4.4 FPN - *Feature Pyramid Network*

A arquitetura FPN (30), originalmente projetada para detecção de objetos, é uma pouco mais complexa que a U-Net. Ainda segundo o princípio de contração e expansão, na FPN a etapa de contração é parecida com a U-Net, com uma sequência de 5 blocos convolucionais ligados por operações de *pooling* máximo. O caminho de expansão é um pouco diferente, acompanhando a Figura 2.13: os blocos *up conv* consistem em camadas de *Upsampling2D*, que expande a escala de uma entrada dado determinado fator (2x ,4x, 8x, 16x), o bloco convolucional simples, a operação de adição a camada de dimensão equivalente compressão, os blocos *up conv* com borda azul, que representam blocos convolucionais duplos após a operação de expansão, o bloco *conv concat* representa uma operação de concatenação entre as 4 camadas superiores seguidos de um bloco convolucional com uma convolução simples de 1 filtro. Todas as camadas possuem filtros com dimensão $[3 \times 3]$

Figura 2.13: Arquitetura FPN.

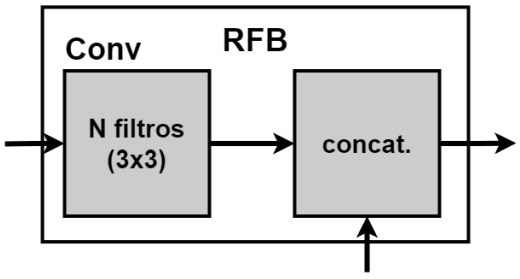


Fonte: Autor

2.4.5 RFB - *Reverse Fusion Block*

RFB é um mecanismo de fusão de características em arquiteturas de aprendizado profundo. Recentemente vem sendo apresentado em alguns trabalhos como solução para refinamento de aprendizado (31, 32). Apresentado na Figura 2.14 esse bloco é composto de uma convolução com n filtros $[3 \times 3]$ seguindo de uma operação de fusão. Esta operação pode ser tanto uma concatenação como uma soma, multiplicação ou até mesmo um produto das entradas.

Figura 2.14: Bloco RFB.



Fonte: Autor

3
Trabalhos relacionados

Trabalhos focados em reconstrução de áudio com fragmentação temporal não são comuns, mesmo que já seja uma tarefa definida na literatura. Contudo, essa tarefa é interpretada em alguns trabalhos como melhoramento de áudio. Alguns trabalhos recentes nesse campo empregam técnicas de aprendizado profundo como *Autoencoders* (33), GANs (*Generative Adversarial Networks*) (34) e LSTMs (*Long-Short Term Memory*) (35) para solucionar problemas parecidos. Deste modo, este capítulo apresentará alguns trabalhos relacionados a tarefa de reconstrução de áudios.

Adler et al. (4) foram os primeiros a definir a tarefa de reconstrução de áudio (*Audio Inpainting*) análoga à de visão computacional, denominada *Image Inpainting*, que em suma, restaura um sinal distorcido ou oculto. Nesse método, as posições das distorções dos áudios são conhecidas e os áudios são representados em quadros sobrepostos no domínio do tempo. Cada quadro possui 64ms e 75% de sobreposição, representados através dos algoritmos DCT (*Discrete Cosine Transform*) e Dicionário de Garbor. Para realização do teste foram definidos três conjuntos, dois de fala com 16kHz e 8kHz e um de música com 16kHz. Cada conjunto com 10 amostras de 5 segundos originados da base SiSEC 2008 (*Signal Separation Evaluation Campaign*) (36), que possui falas de diversos interlocutores e músicas instrumentais. Com uma quantidade de distorções variando de 0.2% a 46% do tamanho do quadro e restauração através da estratégia gulosa do algoritmo OMP (*Orthogonal Matching Pursuit*), foi alcançado 20dB de SNR (*Signal-to-Noise Ratio*), métrica que utiliza o ruído de resultante de uma reconstrução de sinal. Embora só utilizem métodos lineares para a reconstrução, seu trabalho trouxe uma contribuição significativa ao definir um problema antes inexplorado.

Lim et al. (10) apresentam um método de super-resolução para a melhoria da qualidade de banda de áudios instrumentais. Em vias de fato também seria um problema de *Audio Inpainting*, mas como a maioria trata mais da reconstrução temporal, os autores definiram como um problema de super-resolução, definindo o problema como aumento do número de frequências de um espectrograma. Resolvendo-o através de uma rede neural *Autoencoder*, proposta denominada TFNet (*Time-Frequency Network*), nesse método o sinal

é melhorado tanto no domínio do tempo, quanto no domínio das frequências, em ambos os caminhos utilizando o *Autoencoder* AudioUNet (37). Utilizando as bases de fala e de músicas instrumentais, VCTK (38) e PIANO (39), ambas de 16kHz e, respectivamente, 44 e 16 horas de duração. Utilizando a métrica SNR no fator de escala 4x o método alcançou 15dB no VCTK e 23dB no PIANO, no fator 8x os resultados foram 12dB no VTCK e 15dB no PIANO.

Marafioti et al. (8) concentram-se na reconstrução de lacunas temporais de áudio com uma duração fixa de 64 milissegundos. Eles construíram um ambiente controlado para demonstrar que o contexto associado com o sinal perdido facilita o processo de reconstrução. Seu conjunto de dados foi composto exclusivamente de músicas do gênero instrumental da base FMA (*Free Music Archive*), e a abordagem consistiu em extrair características, como o LPC (*Linear Prediction Coding*) e o STFT (*Short-time Fourier Transform*). Para cada 320 ms, foi aplicado uma lacuna de 64 ms no centro do intervalo. Depois disso, um *Autoencoder* atuou como um codificador de contexto, recebendo como entrada as extremidades da faixa para completar as lacunas centrais, considerando apenas o segmento perdido durante a avalição. Finalmente, os autores consideraram apenas a lacuna reconstruída para aplicar as métricas SNR e ODG (*Objective Difference Grades*) (40) que mede a percepção humana de distorção na reconstrução, alcançando uma média de 21dB e -0.8, respectivamente.

Posteriormente, Marafioti et al. (6) apresentaram uma estratégia de *Audio Inpainting* baseada em GAN para restaurar longas lacunas temporais em faixas musicais. A solução proposta, chamada GACELA, considera dois aspectos principais para a reconstrução. Primeiramente, determina cinco discriminadores paralelos para avaliar a reconstrução em cinco escalas diferentes de contexto em relação ao segmento perdido. Em seguida, avalia cada contexto para determinar as variáveis latentes da GAN. Eles realizaram testes com intervalos de 320ms a 1.500ms, ainda posicionados de forma central em um contexto maior. Ainda para a base de dados FMA e considerando apenas a métrica ODG em 64 faixas da base demonstrou média de -0.6 para as lacunas. Entretanto, concluem que os artefatos gerados durante o processo de reconstrução continuam a ser perceptíveis.

Ebner and Eltelt (41) apresentou uma estratégia de reconstrução baseada em GANs para trabalhar com longas lacunas de até 500ms. Em sua abordagem, as lacunas precisam ser centralizadas em um período específico maior do que a região a ser reparada. Eles propõem a utilização Wasserstein GAN (42) com dois discriminadores, para avaliar uma predição de contexto curto e longo. Em seguida, estas predições de contexto são mescladas na tentativa de minimizar a geração de ruído possível para o ouvinte. Para demonstrar seus resultados, eles

usaram faixas de música instrumental de conjuntos PIANO e MAESTRO (43). Por fim, avaliaram a reconstrução usando a métrica ODG em -1.33.

Morrone et al. (7) mostraram que as tarefas de *Audio Inpainting* também podem ser abordadas de forma multimodal. Eles usam recursos de áudio e vídeo concatenado quadro a quadro como entradas para um LSTM bidirecional (BLSTM) empilhada. Seu conjunto de dados possui um conteúdo controlado, com um ator falando para uma câmera posicionada na sua frente. Embora as lacunas a serem preenchidas nesta estratégia tenham sido posicionadas aleatoriamente, é essencial ressaltar que, durante o processo de reconstrução, as posições são conhecidas. Foi mostrado que, quando as lacunas são muito longas, as características do áudio não são suficientes para a reconstrução, exigindo a adição de características visuais.

Na Tabela 3.1 podemos revisar as abordagens e resultados nos trabalhos relacionados. A maioria deles se referenciam entre si e, com a evolução da tarefa, a métrica SNR foi substituída pelo ODG pela facilidade de interpretação. Contrastando com os trabalhos anteriores, neste trabalho buscamos apresentar um método de *Audio Inpainting* que não conhece as posições de lacuna e generalizar o processo através de diferentes gêneros musicais proporcionando maior diversidade de sinais e consistências com um conjunto maior.

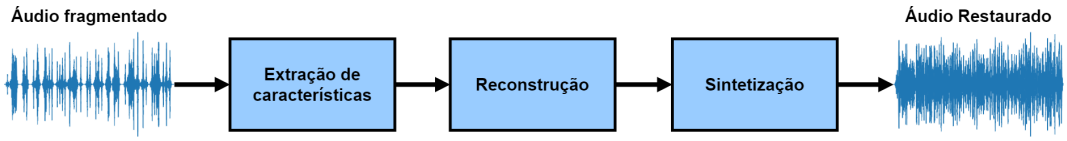
Tabela 3.1: Visão geral dos trabalhos relacionados.

Trabalho	**Abordagem**	**Base**	**SNR**	**ODG**
Adler et al. (4)	Guloso	SiSEC 2008	20dB	-
Lim et al. (10)	Autoencoder	PIANO	23dB	-
Marafioti et al. (8)	Autoencoder	FMA (Instrumental)	21dB	-0.8
Marafioti et al. (6)	GAN	FMA (Instrumental)	-	-0.6
Ebner and Eltelt (41)	GAN	PIANO / MAESTRO	-	-1.33

4
Metologia

Neste Capítulo detalhamos nosso método para reconstrução de frequências perdidas em áudios. Para isso, este método possui três fases principais: extração de características, reconstrução espectral e sintetização do sinal, como ilustrado na Figura 4.1.

Figura 4.1: Fases para reconstrução do áudio.



Fonte: Autor

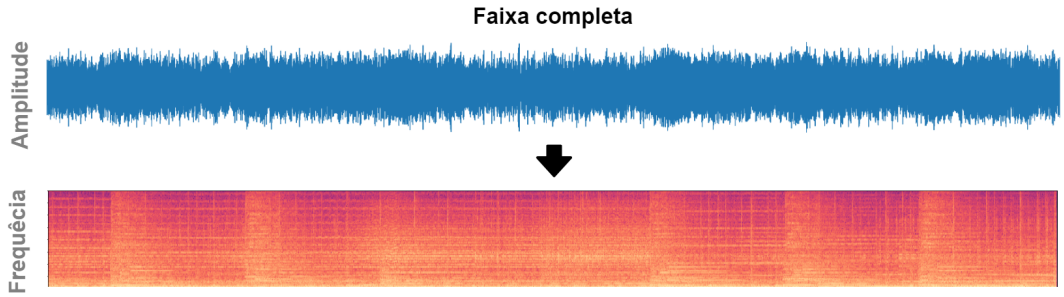
A estrutura desta metodologia visa generalizar a avaliação para qualquer base de *Audio Inpainting* de modo que cada fase possa ser avaliada individual e globalmente.Em suma, a fase "Extração de Características" consiste em transformar um áudio fragmentado em forma de onda em sua equivalência bidimensional de espectrograma, a fase de "Reconstrução" em preencher as lacunas temporais e a fase de "Sintetização" em transformar o espectrograma bidimensional resultante da reconstrução de volta a forma de onda. Cada uma destas fases produzem um campo de estudo próprio como a análise de representações (44) ou diferentes métodos baseados em aprendizado profundo para sintetizar um áudio (45, 46, 47).

4.1
Extração de Características

O objetivo principal do *Audio Inpainting* é tornar o problema de áudio mais parecido possível com seu correspondente em imagem (4). Para isso ser possível é necessário trazer o áudio do formato de onda unidimensional para o formato de espectro bidimensional. Segundo Scaringella et al. (44) em relação a problemas de auto regressão, devemos utilizar características baseadas em timbre, por isso convertendo nossas ondas de entrada 1D em espectrogramas 2D através do algoritmo STFT.

A Figura 4.2 ilustra a transformação de um áudio de 30 segundos e 16KHz no formato de onda para um espectrograma correspondente de dimensões 7500×128, sendo a primeira dimensão representando o domínio do tempo e a segunda representando o domínio das frequências.

Figura 4.2: Transformação de onda em espectrograma.



Fonte: Autor

A construção do espectrograma depende de como está estruturado a relação de tempo e frequência, quanto mais frequência forem requeridas melhor será a representação temporal necessária e vice-versa, chegando ao limite que de uma frequência que representa o áudio original.

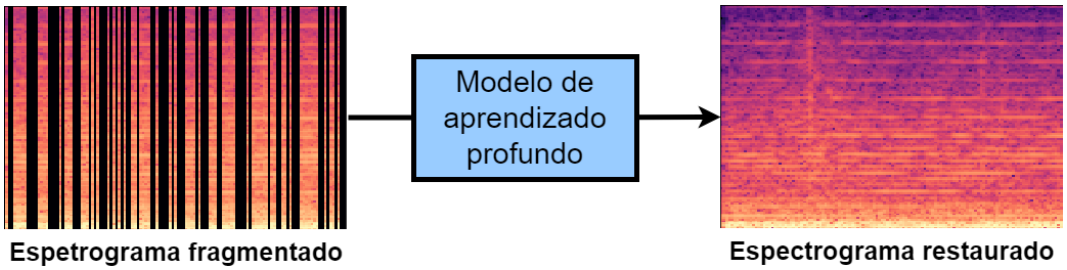
Embora neste estudo tenhamos usado o algoritmo STFT, qualquer método de representação de áudio em duas dimensões pode ser usado nas fases seguintes da metodologia. Contudo, deve-se atentar para dois pontos importantes, primeiro, a representação deve ser inversível para poder retornar ao formato de onda com o mínimo de perda possível. Segundo, deve-se comprovar que o método de reconstrução consiga interpretar semanticamente a construção do espectro.

4.2 Reconstrução

O processo de restauração basea-se na utilização de algoritmo de aprendizado profundo para reconstrução de um espectrograma temporalmente fragmentado. Deste modo, buscando um método mais inteligente de inferir as frequências perdidas indo além do comportamento de simples algoritmos de interpolação clássica.

Assim, como ilustra a Figura 4.3, o modelo de aprendizado profundo escolhido para desempenha esta tarefa pode ser qualquer um que receba uma entrada corrompida e retorne uma saída de mesma dimensão reconstruída. Podendo variar de redes neurais recorrentes como LSTMs, GANs, *Autoencoders* ou até mesmo modelos não lineares como Transformers.

Figura 4.3: Restauração do espectrograma.



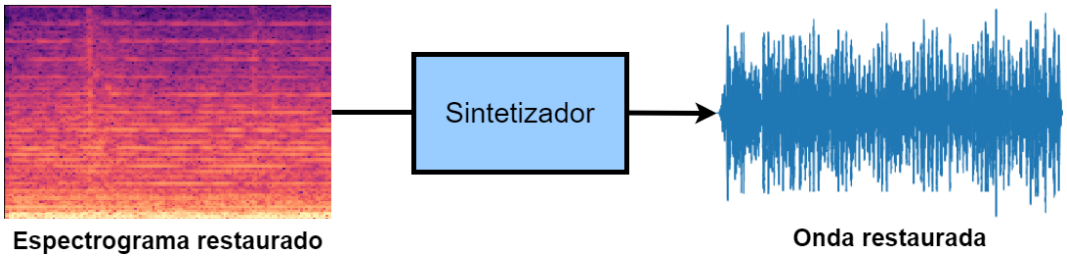
Fonte: Autor

Considerando a maneira que os *Autoencoders* operam, a estratégia de compressão e expansão do mapa de características, intuitivamente é a melhor maneira de suprimir falhas de um sinal. A maioria dos primeiros trabalhos de *Audio Inapainting* que envolvem estratégia de aprendizado profundo também tiveram a mesma iniciativa com os autoencoders. Muito devido à falta de exploração na área, os trabalhos posteriores não exploram tanto o potencial desses algoritmos. Por isso, buscamos demonstrar o real potencial dos *Autoencoders* para a tarefa reconstrução de áudio, indo além de apenas mais um estudo de hipótese, aplicando-os em fragmentações aleatórias mais próximas do mundo real.

4.3 Sintetização

O processo de sintetização, ilustrado na Figura 4.4, pode ser aplicado de modo simples com técnicas de inversão de espectrograma geralmente por transformada inversa, ou por métodos mais sofisticados como aplicação de outro algoritmo de aprendizado profundo para inversão do espectro que não diretamente inversíveis, exemplo de espectros como essa são os Mel-espectrogramas. A rede neural mais sintetização mais conhecida é a Wavenet, nesta rede a onda é melhor interpretada por mecanismos exclusivos da técnica, como a convolução causal.

Figura 4.4: Sintetização do espectrograma.



Fonte: Autor

Neste trabalho utilizamos para extração de características o algoritmo STFT, esse tipo de espectrograma é pseudo-inversível através de um algoritmo iterativo de sintetização chamado Griffin-Lim.

5
Experimento

Neste Capitulo, descrevemos como nosso conjunto de dados foi coletado, como ele é estruturado, como as características foram extraídas, quais as métricas de avaliação foram utilizadas, como é realizado processo de treinamento e seleção dos modelos de aprendizado profundo é guiado pela eficácia na restauração de espectrograma e os resultados finais de percepção do experimento.

5.1
Base de dados

Utilizamos uma parte do conjunto de dados *Free Music Archive* (FMA) (48), esse conjunto foi inicialmente composto para várias tarefas de MIR *(Music Information Retrieval*), que basicamente envolvem classificação e organização de músicas pelos seus metadados como gênero, titulo, artista, etc. Entretanto, perceberam a necessidade de uma base maior no campo de processamento ponta-a-ponta de áudios, gerada pela falta de bases consistentes e genéricas que disponibilizem a mídia em si. Observando pelo ponto de vista da visão computacional é muito fácil encontrar bases grandes e confiáveis para diversas tarefas, como MS-COCO (49), PASCAL-VOC (50), ImageNet (51) e entre outras.

O fato de mídias de músicas estarem mais sujeitas a leis de *copyright*, a produção desses conjuntos é um pouco mais complexo. Desconsiderando a complexidade de armazenamento, a maioria das bases disponibiliza apenas as características já extraídas para dificultar uma possível distribuição.

Além da quantidade de faixas, outros fatores devem ser considerados para determinar a diversidade de sinal de um conjunto de dados de músicas, tais como a quantidade de artistas envolvidos e os gêneros musicais presentes. Podemos observar na Tabela 5.1 uma conferência de bases de músicas comumente citadas, bem como as quantidades de faixas e artistas, o ano de publicação e se disponibiliza as mídias no formato de áudio ou não.

Se tratando de conjuntos de dados de música, é necessário considerar a importância da variedade de gêneros musicais, já que é a maneira mais intuitiva de classificarmos músicas, portanto, é a característica que consideremos como diversidade de sinal. Entretanto, a taxonomia de gêneros musicais é um campo

Tabela 5.1: Conferência entre conjuntos de dados de música.

Conjunto	**Faixas**	**Artistas**	**Ano**	**Áudio**
RWC	465	-	2001	sim
CAL500	500	500	2007	sim
Ballroom	698	-	2004	sim
GTZAN	1.000	300	2002	sim
MusiClef	1.355	218	2012	sim
Artist20	1.413	20	2007	sim
ISMIR2004	1.458	-	2004	sim
Homburg	1.886	1.463	2005	sim
103-Artists	2.445	103	2005	sim
Unique	3.115	3.115	2010	sim
1517-Artists	3.180	1.517	2008	sim
LMD	3.227	-	2007	não
EBallroom	4.180	-	2016	não
USPOP	8.752	400	2003	não
CAL10k	10.271	4.597	2010	não
MagnaTagATune	25.863	230	2009	sim
Codaich	26.420	1.941	2006	não
FMA	106.574	16.341	2017	sim
OMRAS2	152.410	6.938	2009	não
MSD	1.000.000	44.745	2011	não
AudioSet	2.084.320	-	2017	não
AcousticBrainz	2.524.739	-	2017	não

Fonte: Adaptado de Defferrard et al. (48)

muito complexo e subjetivo. Já que diariamente surgem novos gêneros, originais ou derivados. Existe um projeto chamado MusicMap[1] sendo constantemente atualizado como os gêneros musicais conhecidos e suas relações de influência, apresentando mais de 500 gêneros conhecidos. Embora esse projeto seja bastante consistente em sua metodologia a subjetividade dessa taxonomia pode ser percebida como apenas com um gênero, o Internacional que, por definição, são todos os externos a um determinada região. Assim, um gênero que em um lugar teria uma classe própria, em outro pode ser apenas parte de um super grupo.

Na base FMA, a taxonomia dos gêneros foi definida a partir dos metadados anotados para cada faixa, constituindo uma hierarquia de 161 gêneros, destes, 16 como raiz dos demais sub-gêneros. Originalmente o FMA consiste em 343 dias de duração de 106.574 faixas com 16.341 artistas e 14.854 álbuns diferentes. Segundo Scaringella et al. (44), características que definem o timbre são a melhor forma de interpretar áudios para tarefas como classificação e auto regressão. Seguindo esse estudo, a partir de características como espectrograma, os autores do FMA agruparam 8 dos 16 gêneros que julgaram mais representativo s, assim

[1]https://www.musicmap.info/

definindo alguns subconjuntos com faixas de 30 segundos.

Para compor nosso conjunto de dados, selecionamos um subconjunto FMA com 13.583 faixas distribuídas em 16 gêneros musicais. Destes, os oito gêneros mais representativos compõem os conjuntos de treinamento, validação e teste: *Eletronic, Experimental, Rock, Hip-Hop, Folk, Instrumental, Pop, e International*. Os oito gêneros sobressalentes complementam apenas o conjunto de teste, de modo a garantir a abstração do treinamento: *Classical, Historic, Jazz, Country, Soul-RnB, Spoken, Blues, e Easy Listening*. A Tabela 5.2 descreve a distribuição de cada gênero musical nos conjuntos de treinamento, validação e teste. De modo a priorizar o balanceamento os conjuntos de treinamento e validação, o gênero *International* ficou com poucas amostras de teste.

Tabela 5.2: Distribuição dos gêneros musicais nos conjuntos de treino, validação e teste.

Gênero	**Qtd. de Áudios**	**Treino**	**Valid.**	**Teste**
Eletronic	1637	800	200	637
Experimental	1624	800	200	624
Rock	1608	800	200	608
Hip-Hop	1585	800	200	585
Folk	1518	800	200	518
Instrumental	1349	800	200	349
Pop	1186	800	200	186
International	1018	800	200	18
Classical	619	0	0	619
Historic	510	0	0	510
Jazz	384	0	0	384
Country	178	0	0	178
Soul-RnB	154	0	0	154
Spoken	118	0	0	118
Blues	74	0	0	74
Easy Listening	21	0	0	21
Total	13583	6400	1600	5583

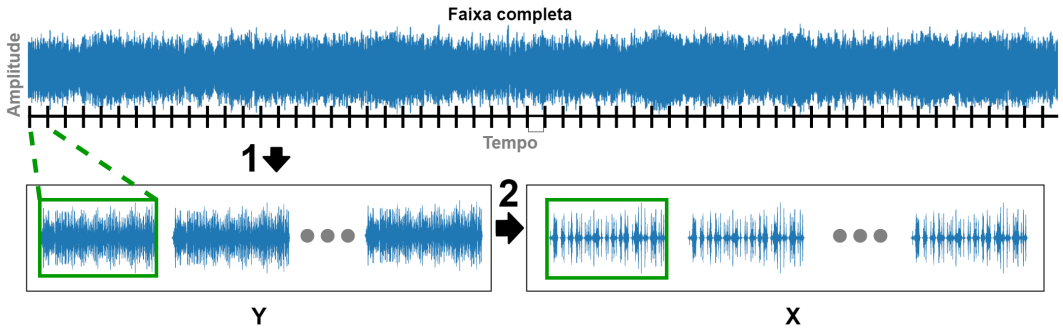
Como alguns desses gêneros são contraintuitivos, vale reespecificá-los como:

- *Experimental*: músicas de produção independente sem um gênero específico.
- *International*: músicas externas às norte-americanas ou europeias.
- *Historic*: músicas do início do século XX.
- *Spoken*: trechos de falas em vídeos.
- *Easy Listening*: sons ambientes e/ou relaxantes.

Originalmente, o conjunto de dados FMA fornece os áudios no formato MP3 estéreo de 44 kHz e bitrate de 320 kbps. Convertemos os áudios para o formato WAV mono de 16 kHz e taxa de bits de 256 kbps. De modo a facilitar a tarefa reduzimos a taxa de amostragem dos áudios para diminuir a complexidade de reconstrução. Desta forma cada áudio de 30s equivale a um vetor inteiro de 16bits com 480.000 posições. Em seguida, cada áudio foi seccionado em 58 partes de 512 ms, descartando os 304ms finais. Foi escolhido 512ms por apresentar dimensão temporal extensa considerando os trabalhos relacionados mais parecidos. Em geral, codecs de transmissão de áudio como o *Opus Codec*,[2] dependendo da largura de banda, transmitem blocos de áudio de 2,5ms a 60ms. Como a perda desses blocos é que causam o efeito que buscamos corrigir, escolhemos 4ms como tamanho padrão de perda. Assim, para cada seção de 512ms foram definidos blocos de 4ms, onde cada seção apresenta um total de 128 blocos.

Feito isso, seguindo uma proporção aleatória, 10% a 70% dos blocos foram substituídos por silêncio (zerar o bin) nas posições selecionadas. Por fim, reunimos os sinais fragmentados finais num novo conjunto de entrada para o problema de *Audio Inpainting*. Em suma, este processo pode ser resumido pela Figura 5.1:

Figura 5.1: Processo de criação de lacunas nos áudios.



Fonte: Autor

1. Seccionamento do áudio (Conjunto referência **Y**);

2. Preenchimento com silêncio em blocos selecionados aleatoriamente (Conjunto de entrada **X**);

Desta feita, dado a distribuição descrita na Tabela 5.1 temos cada conjunto **X** e **Y**:

– 371.200 amostras de treinamento;

[2]https://opus-codec.org/

– 92.800 amostras de validação;

– 323.814 amostras de teste;

Considerando os seguintes argumentos de principais do algoritmo STFT:

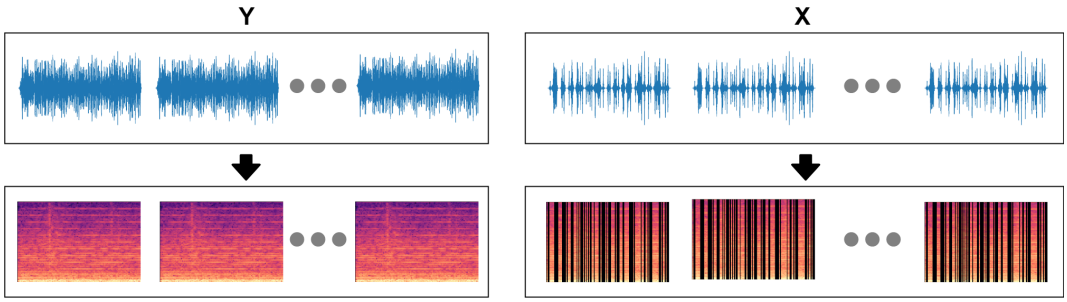
– $frame\_length = 256$;

– $frame\_step = 64$;

– $fft\_length = 255$;

Em uma onda de entrada de 512ms e taxa de amostragem de 16 KHz, tais argumentos geram um espectrograma de dimensão 128×128 ($tempo \times frequências$), onde cada unidade da dimensão temporal representa 4ms da onda original (Figura 5.2).

Figura 5.2: Conjuntos convertidos para espectrogramas.



Fonte: Autor

Nosso conjunto de dados tem algumas vantagens em relação aos anteriormente publicados, como, por exemplo: (a) a quantidade de faixas, com 13.583; (b) a variedade de gêneros musicais; (c) as múltiplas lacunas em diferentes quantidades e posições nas janelas de áudio; (d) a extensão temporal de observação, com 520 ms.

5.2 Métricas

Os modelos foram avaliados em três métricas diferentes: PSNR (*Peak Signal-to-Noise Ratio*), NRMSE (*Normalized Root Mean Square Error*) e ODG (*Objective Difference Grade*). Escolhidas de tal modo que possamos avaliar dois aspectos diferentes, reconstrução e percepção. Usando PSNR e NRMSE para medir a qualidade da restauração do espectrograma, enquanto, através da metodologia PEAQ (*Perceptual Evaluation of Audio Quality*) utilizar o ODG para medir ruído humanamente perceptível na onda resultante.

5.2.1 PSNR

O PSNR é uma métrica utilizada para medir a qualidade da restauração de sinais bastante comum na literatura (52). Medida em escala logarítmica (decibéis), esta métrica segue o comportamento de quanto maior, melhor a restauração do sinal. Em suma, funciona compondo uma relação da potência máxima do sinal em relação à potência do ruído de um sinal de referência. Os primeiros trabalhos de *Audio Inpainting* utilizaram a métrica SNR, definida pela Equação 5-1, contudo, esta não é tão estável em dados muito variados.

Formalmente, podemos definir essa métrica através da Equação 5-2, onde calculamos a qualidade de um sinal restaurado y em relação a um sinal de referência x, MAX é o valor máximo possível alcançado pelo sinal e o MSE (*Mean Square Error*) é o ruído resultante da restauração definido pela Equação 5-3.

$$SNR(x,y) = 10\log_{10}(\frac{x^2}{MSE(x,y)}) \tag{5-1}$$

$$PSNR(x,y) = 10\log_{10}(\frac{MAX^2}{MSE(x,y)}) \tag{5-2}$$

$$MSE(x,y) = \frac{1}{N}\sum_{i=0}^{N}(x-y)^2 \tag{5-3}$$

5.2.2 NRMSE

O NRMSE é outra métrica popular para medir a qualidade da reconstrução do sinal. Em suma, esta métrica basicamente calcula a raiz quadrada do MSE normalizada pela faixa de valores dos dados medidos, esta métrica segue o comportamento de quanto menor, melhor a restauração do sinal. Seja y_{min} o valor mínimo e y_{max} o valor máximo de um sinal restaurado e x um sinal de referência. Formalmente podermos definir através da Equação 5-4.

$$NRMSE(x,y) = \frac{\sqrt{MSE(x,y)}}{y_{max} - y_{min}} \tag{5-4}$$

5.2.3 ODG

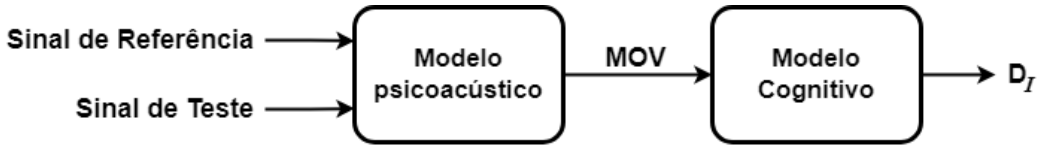
O ODG é uma métrica de qualidade perceptual de áudio que só pode ser calculada através do PEAQ. O PEAQ é um método padronizado que calcula objetivamente a qualidade de reconstrução de um áudio. Para isso é necessário extrair-se 11 características diferentes dos sinais de teste e de referência, em um modelo chamado psicoacústico. Esse modelo busca simular o comportamento do

Tabela 5.3: Interpretação do ODG.

ODG	Descrição
0	Imperceptível
-1	Perceptível, mas não irritante
-2	Levemente irritante
-3	Irritante
-4	Muito irritante

sinal em todas as fases da audição como ouvido externo e interno. A diferença das características de teste e referência produzem 11 variáveis chamadas *MOV* (*Model Output Variables*). Por fim, um modelo cognitivo, sendo basicamente uma rede neural, transforma as *MOV* em um índice de distorção D_I. Esse processo, demonstrado na Figura 5.3, é registrado no padrão ITU-R BS.1387,[3] e os pesos do modelo cognitivo original são patenteados e não podem ser utilizados livremente. Contudo, utilizamos o modelo criado por Kabal et al. (40), que efetua uma engenharia reversa no padrão para definir os pesos do modelo cognitivo aproximados do original e disponibilizá-los.

Figura 5.3: Fluxo de execução PEAQ.



Fonte: Autor

Obtidos os índices de distorção D_I podemos calcular o ODG através da Equação 5-5, onde *sig* representa uma função sigmoide e os parâmetros $b_min = -3,98$ e $b_{max} = 0,22$. Ambos os parâmetros são definidos por Kabal et al..

$$ODG = b_{min} + (b_{max} - b_{min})sig(D_I) \quad (5\text{-}5)$$

Por fim, o ODG será um valor entre -4 e 0 que pode ser interpretado em escalas como descrito na Tabela 5.3.

5.3 Configurações

Nossos modelos foram treinadas usando uma CPU i7 3,40 GHz de oito núcleos com uma GPU NVIDIA TESLA K80. O treinamento foi baseado na otimização Adam (53) com momento de 0,999, decaimento exponencial de 0,9 e epsilon de 1e-07, normalização de lote com decaimento de 0,9997 e epsilon de 0,001 com uma taxa de aprendizado fixa de 0,001, e MSE como função

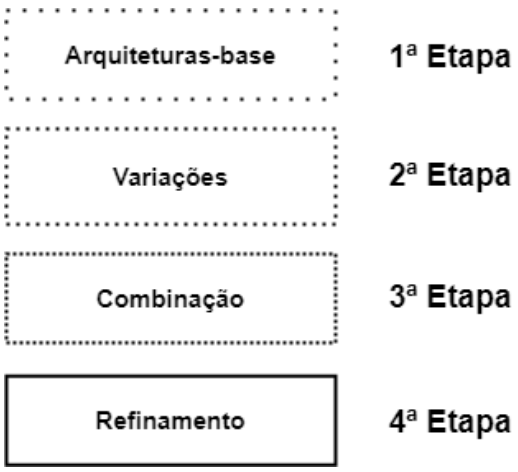
[3]https://www.itu.int/rec/R-REC-BS.1387

de custo. Os pesos da rede foram inicializados com Glorot (54) com semente zero. Normalizamos nosso conjunto de dados e executamos nossos experimentos por no máximo 100 épocas e com *batchsize* 58. Para cada época foi feito uma validação, visando preservar a época com as maiores métricas de reconstrução.

5.4 Treinamentos

Esta Seção descreve cada experimento realizado através de um fluxo de treinamento de diversos modelos. Para definir a melhor arquitetura de reconstrução dos espectrogramas foram realizados treinamentos em um esquema de quatro etapas: a primeira define arquiteturas-base do fluxo; a segunda define as principais variações das arquiteturas iniciais selecionadas para este experimento; a terceira realiza combinações entre as variações, utilizando apenas os melhores modelos da etapa anterior. Por fim, a quarta consiste em refinamentos do melhor modelo da etapa de combinação. Como ilustra a Figura 5.4.

Figura 5.4: Etapa de treinamentos para seleção dos modelos.



Fonte: Autor

As arquiteturas-base utilizadas na primeira etapa foram os *Autoencoders* U-Net e FPN, modelos clássicos na literatura. A ideia é mostrar que arquiteturas clássicas conseguem aprender a tarefa. Assim, a Tabela 5.4 ilustra os resultados de validação alcançados nessa primeira fase de treinamento.

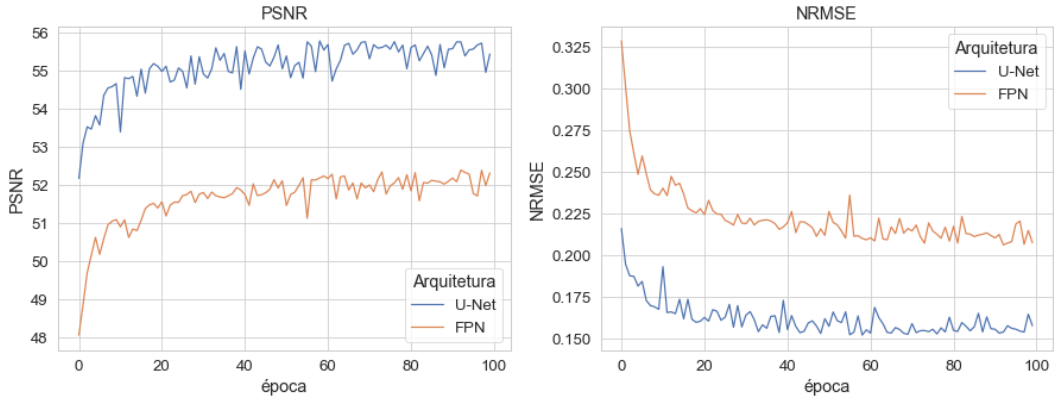
Durante o treinamento, a U-Net alcançou o melhor resultado de validação na época 58 e a FPN na época 92. Como ilustra o gráfico de convergência na Figura 5.5, ambas as métricas de reconstrução constatam que a arquitetura U-Net tem um desempenho melhor que a FPN, alcançando os resultados 55,7687 dB de PSNR e 0,1523 de NRMSE. Contudo, nesta etapa ainda não realizamos descartes de arquiteturas, tais descartes são realizados na etapa de variações.

Tabela 5.4: Resultados de reconstrução do conjunto de validação com os modelos da primeira etapa.

	PSNR▲		**NRMSE▼**	
Gênero	**U-Net**	**FPN**	**U-Net**	**FPN**
Electronic	**53.6760**	50.3287	**0.1540**	0.2068
Experimental	**57.3531**	54.0423	**0.1579**	0.2118
Rock	**52.6030**	49.3764	**0.1705**	0.2288
Hip-Hop	**52.9716**	49.6591	**0.1531**	0.2050
Folk	**58.0909**	54.4880	**0.1395**	0.1951
Instrumental	**59.0135**	55.4977	**0.1454**	0.2010
Pop	**54.9014**	51.5697	**0.1556**	0.2101
International	**57.5399**	54.1499	**0.1420**	0.1929
Média	**55.7687**	52.3890	**0.1523**	0.2064

Porém, o melhor modelo da primeira etapa gera mais variações de teste por se mostrar mais promissor.

Figura 5.5: Convergência de validação das arquiteturas da 1ª etapa.

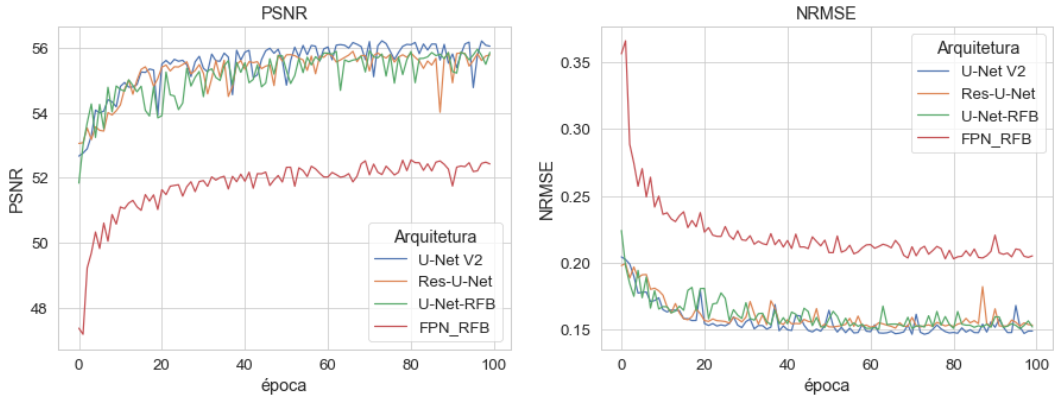


Fonte: Autor

Na etapa de variações foram selecionados 3 variações para a base U-Net e uma para a base FPN. Das variações da U-Net foram escolhidas as arquiteturas U-Net V2, Res-U-Net e U-Net-RFB. Inspirado no trabalho de Busson et al. (31), utilizamos os blocos RFB como método de fusão para a arquitetura U-Net e FPN. Respectivamente, as arquiteturas alcançaram os melhores resultados de validação nas épocas 73, 66 e 96. Já para arquitetura FPN a única variação treinada foi a FPN-RFB e alcançou o melhor resultado na época 80, como ilustra o gráfico de convergência na Figura 5.6.

Os resultados de validação da Tabela 5.5 mostram que o melhor resultado alcançado na segunda etapa de treinamento foi obtido utilizando a variação U-Net V2, com 56,2264 dB de PSNR e 0,1465 de NRMSE. Nesta etapa todas as variações alcançaram resultados melhores que suas respectivas arquiteturas

Figura 5.6: Convergência de validação das arquiteturas da 2ª etapa.



Fonte: Autor

base. Contudo, nesta etapa as arquiteturas com os piores resultados foram descartadas, no caso, ambos os modelos FPN.

Tabela 5.5: Resultados de reconstrução do conjunto de validação com os modelos da segunda etapa.

Gênero	PSNR▲				NRMSE▼			
	U-Net V2	**Res-U-Net**	**U-Net-RFB**	**FPN-RFB**	**U-Net V2**	**Res-U-Net**	**U-Net-RFB**	**FPN-RFB**
Electronic	**54,1059**	53,7744	53,8189	50,5222	**0,1488**	0,1534	0,1534	0,2028
Experimental	**57,7869**	57,5243	57,5892	54,1726	**0,1524**	0,1563	0,1557	0,2090
Rock	**52,9829**	52,6776	52,7581	49,4940	**0,1651**	0,1697	0,1692	0,2259
Hip-Hop	**53,3412**	53,0586	53,0556	49,8008	**0,1485**	0,1527	0,1532	0,2016
Folk	**58,6509**	58,2547	58,3701	54,6845	**0,1326**	0,1378	0,1368	0,1909
Instrumental	**59,5594**	59,1736	59,2684	55,7079	**0,1386**	0,1438	0,1430	0,1966
Pop	**55,3386**	54,9985	55,0780	51,7244	**0,1499**	0,1547	0,1542	0,2067
International	**58,0458**	57,7538	57,8206	54,3058	**0,1361**	0,1403	0,1397	0,1898
Média	**56,2264**	55,9020	55,9699	52,5515	**0,1465**	0,1511	0,1506	0,2029

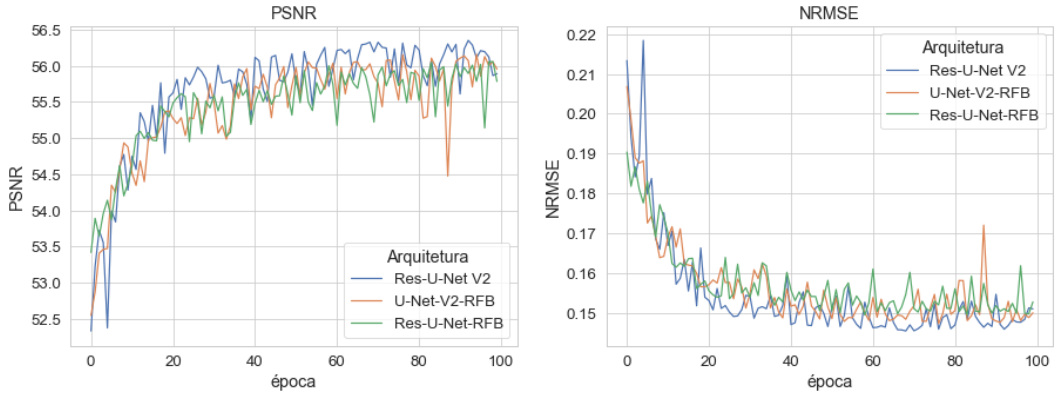
Na etapa de combinação as variações com os melhores resultado foram combinadas em três arquiteturas a Res-U-Net V2, o U-Net-V2-RFB e a Res-U-Net-RFB. Cada uma, respectivamente, alcançou os melhores resultados nas épocas 92, 94, 98. Como exposto na Tabela 5.6 e no gráfico de convergência da Figura 5.7 a arquitetura que alcançou os melhores resultados foi a Res-U-Net V2, com 56,3534 dB de PSNR e 0,146 de NRMSE. Assim, é escolhido apenas o melhor modelo para a etapa final de refinamento.

Por fim, na etapa de refinamento construímos um modelo duplo a partir do melhor modelos da etapa de combinação. Este modelo tem por objetivo adicionar mais um estágio no processo de reconstrução do espectrograma, tendo a primeira ocorrência de modelo para o processo de reconstrução bruto e a segunda ocorrência para refinamento. Desta feita, construímos o modelo Dual-Res-U-Net V2, representado na Figura 5.8. Por conseguinte, foram realizados dois métodos

Tabela 5.6: Resultados de reconstrução do conjunto de validação com os modelos da terceira etapa.

Gênero	PSNR▲			NRMSE▼		
	Res-U-Net V2	**U-Net-V2-RFB**	**Res-U-Net-RFB**	**Res-U-Net V2**	**U-Net-V2-RFB**	**Res-U-Net-RFB**
Electronic	**54,2169**	54,0255	53,8998	**0,1486**	0,1505	0,1528
Experimental	**57,8991**	57,7258	57,7087	**0,1520**	0,1535	0,1546
Rock	**53,0700**	52,9036	52,8086	**0,1650**	0,1669	0,1686
Hip-Hop	**53,4567**	53,2201	53,1423	**0,1486**	0,1509	0,1528
Folk	**58,8289**	58,6323	58,4541	**0,1313**	0,1332	0,1360
Instrumental	**59,7195**	59,5141	59,3840	**0,1374**	0,1394	0,1418
Pop	**55,4537**	55,2739	55,1464	**0,1495**	0,1513	0,1535
International	**58,1821**	57,9776	57,9390	**0,1356**	0,1373	0,1388
Média	**56,3534**	56,1591	56,0604	**0,1460**	0,1479	0,1498

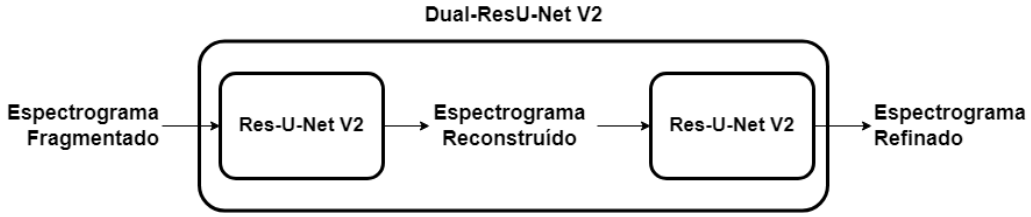
Figura 5.7: Convergência de validação das arquiteturas da 3ª etapa.



Fonte: Autor

de treinamento, onde o primeiro treina o modelo do zero, porém com dobro de épocas e o segundo que reaproveita os pesos da etapa de treinamentos anterior e treina apenas a composição de refinamento, denominando este novo modelo como Dual-Res-U-Net V2 Transfer por conter a estratégia de transferência de aprendizado.

Figura 5.8: Modelo de refinamento Dual-Res-U-Net V2.



Fonte: Autor

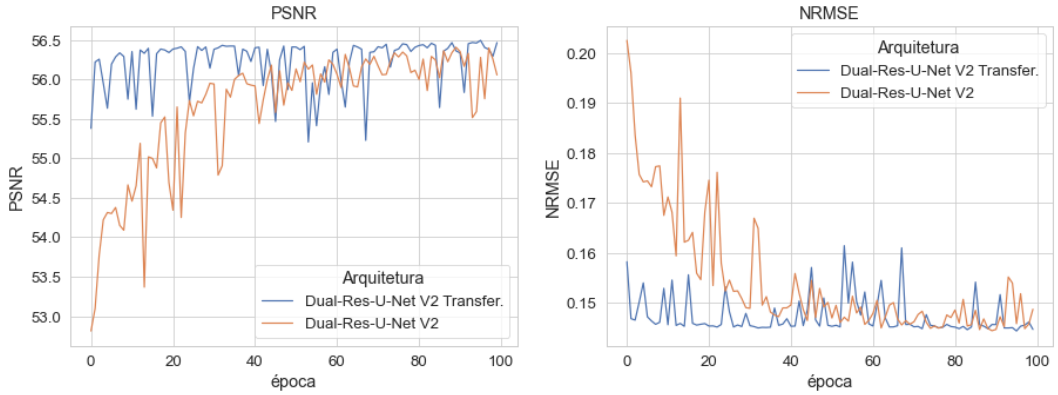
Assim, como expõe a Tabela 5.7 as maiores médias alcançadas foram pela

estratégia de transferência de aprendizado, alcançando 56,5925 dB de PSNR e embora os melhores resultados de NRMSE tenham variado entre modelos a melhor média continuou com o modelo Dual-ResU-Net V2 Transfer, chegando a 0,1444. Também, dominou os melhores resultados durante todo o treinamento, como exposto na Figura 5.9.

Tabela 5.7: Resultados de reconstrução do conjunto de validação com os modelos da quarta etapa.

	PSNR▲		**NRMSE▼**	
Gênero	**Dual-Res-U-Net V2**	**Dual-Res-U-Net V2 Transfer.**	**Dual-Res-U-Net V2**	**Dual-Res-U-Net V2 Transfer.**
Electronic	54,3073	**54,3522**	**0,1467**	0,1470
Experimental	57,9472	**58,0737**	0,1505	**0,1500**
Rock	53,1454	**53,1754**	**0,1629**	0,1634
Hip-Hop	53,5458	**53,5601**	**0,1466**	0,1478
Folk	58,8430	**58,9763**	0,1304	**0,1295**
Instrumental	59,7414	**59,8836**	0,1365	**0,1356**
Pop	55,5057	**55,5805**	**0,1480**	**0,1480**
International	58,2055	**58,3385**	0,1342	**0,1339**
Média	56,4052	**56,4925**	0,1445	**0,1444**

Figura 5.9: Convergência de validação das arquiteturas da 4ª etapa.



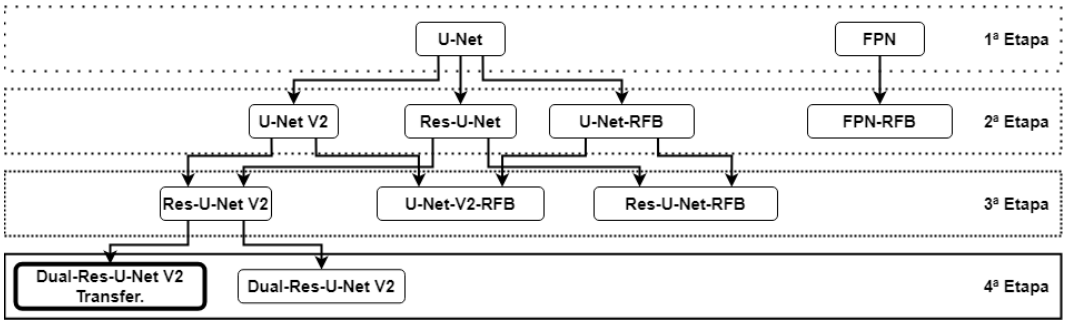
Fonte: Autor

Ao fim, podemos representar o processo através da Figura 5.10, que apresenta todas as arquiteturas experimentadas em suas respectivas etapas, também representado suas relações de dependências.

5.5 Resultados

Após a fase treinamentos temos a classificação geral das arquiteturas e suas respectivas etapas descrito na Tabela 5.8. Começando a alisar esta classificação, à primeira vista podemos perceber que a arquitetura FPN e

Figura 5.10: Fluxo de seleção de modelo em etapas.



Fonte: Autor

sua variante ficaram em último em suas respectivas etapas. Considerando as semelhanças de arquitetura com a U-Net na fase de contração, podemos concluir que as FPNs, em suas fases de expansão, não se adaptam muito bem ao problema de reconstrução de espectrogramas.

Contudo, considerando os modelos da segunda etapa, existe uma clara disparidade de resultados entre a arquitetura U-Net V2 e os demais. Embora todas as variantes da U-Net tenham apresentado resultados melhores que o modelo original, é claro o impacto que a abolição de operações de *pooling* causam na tarefa. Esse efeito ocorre pela característica do sinal de entrada, em espectrogramas às relações de vizinhança dos valores tem mais valor semântico que em imagens, então, estratégias de representação por região como *pooling* resultam em mais perda semântica no espectrograma. O impacto desse efeito fez com que a arquitetura U-Net V2 desempenhasse melhor que a maioria dos modelos da etapa de combinação.

Por conseguinte, se consideramos todas as arquiteturas da primeira até a terceira etapa a que obteve o melhor desempenho foi a Res-U-Net V2. Embora as variantes que adotam os blocos RFB tenham superado todas suas predecessoras, quando combinada com a estratégia de convolução espapaçada da U-Net V2 e o mecanismo GRC da Res-U-Net, não alcançaram os resultados da Res-U-Net V2.

Todavia, acreditando que poderia extrair um pouco mais de desempenho do melhor modelo, a etapa de refinamento visa suavizar os artefatos de construção resultante de todo *Autoencoder*. Esses artefatos são ruídos originados do processo de expansão. Então a arquitetura Dual-Res-U-Net V2 pussui mais uma camada de aprendizado para tratar essa característica, alcançando os melhores resultados nos experimentos de validação.

Além disso, os gêneros que possuem os resultados mais baixos de reconstrução são *Eletronic*, *Hip-Hop* e *Rock*. O que esses gêneros têm em comum é a presença predominante de sons produzidos por sintetizadores e

Tabela 5.8: Classificação das arquiteturas.

Posição	**Arquitetura**	**Etapa**
1º	Dual-ResU-Net V2 Transfer.	4
2º	Dual-ResU-Net V2	4
3º	Res-U-Net V2	3
4º	U-Net V2	2
5º	U-Net-V2-RFB	3
6º	Res-U-Net-RFB	3
7º	U-Net-RFB	3
8º	Res-U-Net	2
9º	U-Net	1
10º	FPN-RFB	2
11º	FPN	1

instrumentos eletrônicos. Essa característica faz com que a variação do sinal seja um pouco caótica em relação aos demais gêneros com predominância de sons acústicos, que produzem sons mais uniformes. Assim, a reconstrução do espectrograma fica mais simples em alguns gêneros que em outros.

Encontrada a melhor arquitetura, passamos para a fase de teste. Nesta fase, além da avaliação de reconstrução também consideramos a percepção do ruído presente após a sintetização do sinal. Também, nesse conjunto consideramos todos os 16 gêneros musicais. Evidenciando os valores antes e depois, além do ganho obtido pelo método de reconstrução, como mostrado na Tabela 5.9.

Tabela 5.9: Resultado do conjunto de teste com o modelo Dual-Res-U-Net V2 Transfer.

	Antes			**Depois**			**Ganho**		
Gênero	**PSNR▲**	**NRMSE▼**	**ODG▲**	**PSNR▲**	**NRMSE▼**	**ODG▲**	**PSNR▲**	**NRMSE▼**	**ODG▲**
Electronic	48,7529	0,5300	-3,1209	54,4588	0,1498	-1,7644	+12,9941	-0,4347	+1,3565
Experimental	45,2932	0,5611	-3,2104	58,5276	0,1503	-2,0238	+12,3366	-0,4037	+1,1867
Rock	44,3598	0,5655	-3,2934	53,6197	0,1644	-1,8413	+11,8186	-0,4186	+1,4521
Hip-Hop	43,3860	0,5770	-3,0051	53,8873	0,1479	-1,5708	+13,0807	-0,4367	+1,4343
Folk	42,9498	0,5784	-3,3643	59,5749	0,1287	-2,0042	+13,8635	-0,4415	+1,3601
Instrumental	41,4648	0,5845	-3,3258	58,7827	0,1413	-2,1085	+13,1437	-0,4275	+1,2173
Pop	40,8066	0,5846	-3,2307	56,0634	0,1495	-1,8365	+12,6774	-0,4275	+1,3942
International	41,8011	0,5831	-3,0897	62,1463	0,1218	-1,8611	+13,3934	-0,4082	+1,2286
Classical	40,7757	0,5883	-3,3741	68,8527	0,1087	-2,5840	+13,6780	-0,3844	+0,7901
Historic	55,1747	0,4930	-3,0683	64,3201	0,1131	-1,8888	+14,8651	-0,4417	+1,1795
Jazz	52,3546	0,5097	-3,2785	60,5204	0,1308	-1,9845	+13,5733	-0,4276	+1,2940
Country	49,4550	0,5548	-3,2508	55,6442	0,1482	-1,7864	+12,6944	-0,4303	+1,4644
Soul-RnB	46,9471	0,5584	-3,1631	53,5132	0,1543	-1,6562	+12,7375	-0,4340	+1,5069
Spoken	45,7114	0,5702	-3,1698	65,3030	0,1206	-1,9472	+12,9484	-0,3891	+1,2226
Blues	45,6390	0,5687	-3,1501	58,3804	0,1383	-1,6722	+13,0873	-0,4228	+1,4779
Easy Listening	46,1910	0,5540	-3,0700	57,6062	0,1373	-1,8146	+13,2464	-0,4282	+1,2554
Média	45,6914	0,5601	-3,2029	**58,8251**	**0,1378**	**-1,9053**	**+13,1336**	**-0,4223**	**+1,2976**

Observando os resultados podemos perceber um fato, a princípio, contraintuitivo nas relações dos resultados de reconstrução (PSNR e NRMSE) e de percepção (ODG), onde os gêneros de influência acústica tiveram resultado bons de reconstrução e ruins de percepção em relação aos demais. Isso acontece pela própria característica dos sinais. por exemplo, no gênero *Classical*, onde

naturalmente só há sinais acústicos um ruído leve tem mais impacto na percepção do que um ruído de mesma intensidade em um sinal mais caótico como no gênero *Rock*. Por isso, os ganhos de reconstrução podem variar de escala entre os gêneros.

Por outro lado, também podemos notar a uniformidade de aprendizado do modelo, mesmo em gêneros que nunca foram observados no processo de treinamento. Em todos os gêneros de teste houve estabilidade no ganho, inclusive nos nunca observados pelo modelo. Mantendo os valores próximos e alcançando uma média de 58,8251 dB de PSNR, 0,1378 de NRMSE e -1,9053 de ODG, o que conforme a tabela descrição do ODG significa que o áudio resultante produz um ruído perceptível, mas não irritante em relação ao áudio original. Com ganhos estáveis de +13,1336 dB de PSNR, -0,4223 de NRMSE e +1,2976 da escala ODG.

Por fim, embora a comparação direta entre os trabalhos relacionados não seja a ideal, selecionamos os dois mais trabalhos parecidos para apresentar uma comparação de resultados, exposto na Tabela 5.10. Onde o trabalho do Adler et al. (4) possui a mesma lógica de fragmentação em uma base instrumental e o trabalho do Marafioti et al. (8) aplicado na mesma base, porém com lógica de fragmentação diferente. Nosso trabalho se mostra equiparável na métrica SNR, contudo devido ao método de fragmentação diferente não é comparável pelo ODG.

Tabela 5.10: Comparação com trabalhos relacionados.

Trabalho	**Abordagem**	**Base**	**SNR**	**ODG**
Adler et al. (4)	Guloso	SiSEC 2008	20dB	-
Marafioti et al. (8)	Autoencoder	FMA (Instrumental)	21dB	**-0.8**
Este trabalho	Autoencoder	FMA (Instrumental)	**27dB**	-2,11

6
Conclusão

Neste trabalho, propusemos um método baseado em aprendizado profundo para melhorar a qualidade de áudios de música através da restauração de lacunas perdidas, evidenciando o uso de *Autoencoders*, que aprende a suprimir as falhas em um espectrograma através do processo de compressão e expansão. Para isso, construímos um conjunto de dados de músicas temporalmente degradadas de diversos gêneros musicais, baseado no FMA. Nosso conjunto de dados tem vantagens sobre os anteriormente publicados para mesma tarefa, principalmente devido ao número de faixas, a variedade de gêneros musicais e na posição, além do tamanho dos cortes nos áudios.

Por conseguinte, realizamos um esquema de treinamento em etapas para selecionar o melhor modelo de reconstrução baseado nas métricas PSNR e NRMSE. Dentre os onze experimentos de treinamento a arquitetura que apresentou o melhor resultado de foi a Dual-Res-U-Net V2 com transferência de aprendizado. Desta feita, através do experimento de teste mostramos que nosso método melhorou a qualidade de todos os gêneros musicais, obtendo uma média de ganho de +13,1 dB de PSNR, -0.42 de NRMSE e +1,3 de ODG. Também alcançando resultados equiparáveis aos trabalhos relacionados mais semelhantes.

Em suma, concluímos as seguintes contribuições deste trabalho:

- Construção de uma base de dados robusta para a tarefa de *Audio Inpainting*, de modo a incentivar a pesquisa no campo;
- Definição de um método simples de reconstrução, onde cada etapa pode ser avaliada de modo independente;
- Definição de um método de avaliação claro para ser facilmente reproduzível e comparável, evidenciando a importância da separação de gêneros musicais para avaliação;
- Implementação de um modelo genérico de reconstrução.

Durante a produção desde trabalho gerou-se uma publicação no simpósio brasileiro em multimídia WebMedia 2021 (55), onde apresentamos resultados alcançamos até então.

Como trabalhos futuros, pretendemos a considerar um estudo focado apenas nas representações de áudio utilizadas, por exemplo, buscar representações baseadas em ritmo ou harmonia, pouco além das características baseadas em timbre. Outrossim, visamos o estudo em novos arquiteturas de *Autoencoder* como Vision Transformer (56), assim como refinar a sintetização de áudio utilizando a rede Wavenet (46). Por fim, experimentar a avaliação de treinamento baseado em métricas de percepção.

Referências bibliográficas

[1] Šerif Pilipović, "Discord statistics and facts 2022," Mar. 2022. [Online]. Available: https://levvvel.com/discord-statistics-and-facts/

[2] Samantha, "Opus: One codec to rule them all?" Mar. 2022. [Online]. Available: https://www.onsip.com/voip-resources/voip-fundamentals/opus-one-codec-to-rule-them-all

[3] Troubleshooter.com, "Problem: Packet loss concealment," Mar. 2022. [Online]. Available: https://www.voiptroubleshooter.com/problems/plc.html

[4] A. Adler, V. Emiya, M. G. Jafari, R. Elad, Michael anGribonval, and M. D. Plumbley, "Audio inpainting," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 3, pp. 922–932, 2011.

[5] M. Kegler, P. Beckmann, and M. Cernak, "Deep speech inpainting of time-frequency masks," *INTERSPEECH 2020*, 2020.

[6] A. Marafioti, P. Majdak, N. Holighaus, and N. Perraudin, "Gacela-a generative adversarial context encoder for long audio inpainting of music," *IEEE Journal of Selected Topics in Signal Processing*, 2020.

[7] G. Morrone, D. Michelsanti, Z.-H. Tan, and J. Jensen, "Audio-visual speech inpainting with deep learning," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6653–6657.

[8] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "A context encoder for audio inpainting," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 27, no. 12, p. 2362–2372, Dec. 2019. [Online]. Available: https://doi.org/10.1109/TASLP.2019.2947232

[9] A. Marafioti, N. Holighaus, P. Majdak, N. Perraudin *et al.*, "Audio inpainting of music by means of neural networks," in *Audio Engineering Society Convention 146*. Audio Engineering Society, 2019.

[10] T. Y. Lim, R. A. Yeh, Y. Xu, M. N. Do, and M. Hasegawa-Johnson, "Time-frequency networks for audio super-resolution," in *2018 IEEE International*

Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 646–650.

[11] R. R. Devi and D. Pugazhenthi, "Ideal sampling rate to reduce distortion in audio steganography," *Procedia Computer Science*, vol. 85, pp. 418–424, 2016.

[12] H. Landau, "Sampling, data transmission, and the nyquist rate," *Proceedings of the IEEE*, vol. 55, no. 10, pp. 1701–1706, 1967.

[13] R. F. Rangel, "Transformada de fourier em visão computacional," Mar 2021. [Online]. Available: https://medium.com/turing-talks/transformada-de-fourier-b1775e891cc5

[14] I. N. Sneddon, *Fourier transforms*. Courier Corporation, 1995.

[15] G. B. Folland, *Fourier Analysis and Its Applications*. Florence, KY: Brooks/Cole, 1992.

[16] N. Kehtarnavaz, "Chapter 7 - frequency domain processing," in *Digital Signal Processing System Design (Second Edition)*, second edition ed., N. Kehtarnavaz, Ed. Burlington: Academic Press, 2008, pp. 175–196. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123744906000076

[17] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 32, no. 2, pp. 236–243, 1984.

[18] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio *et al.*, "Tacotron: Towards end-to-end speech synthesis," *arXiv preprint arXiv:1703.10135*, 2017.

[19] A. Böttcher and D. Wenzel, "The frobenius norm and the commutator," *Linear algebra and its applications*, vol. 429, no. 8-9, pp. 1864–1885, 2008.

[20] F. Rosenblatt, "Principles of neurodynamics. perceptrons and the theory of brain mechanisms," Cornell Aeronautical Lab Inc, Buffalo, NY, Tech. Rep., 1961.

[21] Arc, "The perceptron," Jul 2018. [Online]. Available: https://towardsdatascience.com/the-perceptron-3af34c84838c

[22] J. Brownlee, "Gradient descent for machine learning," Mar. 2016. [Online]. Available: https://machinelearningmastery.com/gradient-descent-for-machine-learning/

[23] A. H. Reynolds, "Convolutional neural networks (cnns)," Dez. 2019. [Online]. Available: https://anhreynolds.com/blogs/cnn.html

[24] N. Tomar. Introduction to autoencoders. [Online]. Available: https://medium.com/swlh/introduction-to-autoencoders-56e5d60dad7f

[25] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[26] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/905056c1ac1dad141560467e0a99e1cf-Paper.pdf

[27] A. J. G. Busson, P. R. C. Mendes, D. de S. Moraes, Á. M. G. da Veiga, S. Colcher, and Á. L. V. Guedes, "Decoder-side quality enhancement of jpeg images using deep learning-based prediction models for quantized dct coefficients," in *Proceedings of the Brazilian Symposium on Multimedia and the Web*, 2020, pp. 129–136.

[28] S. Li, F. He, B. Du, L. Zhang, Y. Xu, and D. Tao, "Fast spatio-temporal residual network for video super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 522–10 531.

[29] B. Park, S. Yu, and J. Jeong, "Densely connected hierarchical network for image denoising," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[30] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[31] A. J. G. Busson, "A self-supervised method for blind denoising of seismic shot gathers," Ph.D. dissertation, Pontifical Catholic University of Rio de Janeiro, 2022, https://doi.org/10.17771/PUCRio.acad.59152.

[32] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta, "Beyond skip connections: Top-down modulation for object detection."

[33] X. Lu, Y. Tsao, S. Matsuda, and C. Hori, "Speech enhancement based on deep denoising autoencoder." in *Interspeech*, vol. 2013, 2013, pp. 436–440.

[34] J. Lin, S. Niu, A. J. van Wijngaarden, J. L. McClendon, M. C. Smith, and K.-C. Wang, "Improved speech enhancement using a time-domain gan with mask learning." in *INTERSPEECH*, 2020, pp. 3286–3290.

[35] J. Deng, B. Schuller, F. Eyben, D. Schuller, Z. Zhang, H. Francois, and E. Oh, "Exploiting time-frequency patterns with lstm-rnns for low-bitrate audio restoration," *Neural Computing and Applications*, vol. 32, no. 4, pp. 1095–1107, 2020.

[36] E. Vincent, S. Araki, and P. Bofill, "The 2008 signal separation evaluation campaign: A community-based approach to large-scale evaluation," in *International Conference on Independent Component Analysis and Signal Separation*. Springer, 2009, pp. 734–741.

[37] V. Kuleshov, S. Z. Enam, and S. Ermon, "Audio super resolution using neural networks," *arXiv preprint arXiv:1708.00853*, 2017.

[38] C. Veaux, J. Yamagishi, K. MacDonald *et al.*, "Superseded-cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit," 2017.

[39] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "Samplernn: An unconditional end-to-end neural audio generation model," *arXiv e-prints*, pp. arXiv–1612, 2016.

[40] P. Kabal *et al.*, "An examination and interpretation of itu-r bs. 1387: Perceptual evaluation of audio quality," *TSP Lab Technical Report, Dept. Electrical & Computer Engineering, McGill University*, pp. 1–89, 2002.

[41] P. P. Ebner and A. Eltelt, "Audio inpainting with generative adversarial network," *arXiv preprint arXiv:2003.07704*, 2020.

[42] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and

Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 214–223. [Online]. Available: https://proceedings.mlr.press/v70/arjovsky17a.html

[43] C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C.-Z. A. Huang, S. Dieleman, E. Elsen, J. Engel, and D. Eck, "Enabling factorized piano music modeling and generation with the MAESTRO dataset," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=r1lYRjC9F7

[44] N. Scaringella, G. Zoia, and D. Mlynek, "Automatic genre classification of music content: a survey," *IEEE Signal Processing Magazine*, vol. 23, no. 2, pp. 133–141, 2006.

[45] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous, "Tacotron: Towards end-to-end speech synthesis," 2017. [Online]. Available: https://arxiv.org/abs/1703.10135

[46] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.

[47] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[48] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "FMA: A dataset for music analysis," in *18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017. [Online]. Available: https://arxiv.org/abs/1612.01840

[49] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[50] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[51] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet

Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[52] N. Instruments, "Peak signal-to-noise ratio as an image quality metric," 2013.

[53] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[54] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 315–323.

[55] A. C. Serra, A. J. G. Busson, Á. L. Guedes, and S. Colcher, "Quality enhancement of highly degraded music using deep learning-based prediction models for lost frequencies," in *Anais do XXVII Simpósio Brasileiro de Sistemas Multimídia e Web*. SBC, 2021, pp. 205–211.

[56] Z. Zhang, T. Li, X. Tang, X. Hu, and Y. Peng, "Caevt: Convolutional autoencoder meets lightweight vision transformer for hyperspectral image classification," *Sensors*, vol. 22, no. 10, p. 3902, 2022.