Universidade Federal de Mato Grosso do Sul

Campus de Três Lagoas

Curso de Sistemas de Informação

João Vitor Sartoreto

INTEGRAÇÃO DOS COEFICIENTES CEPSTRAIS DE FREQUÊNCIA MEL NO FRAMEWORK MIR_REF

Três Lagoas/2025

João Vitor Sartoreto

INTEGRAÇÃO DOS COEFICIENTES CEPSTRAIS DE FREQUÊNCIA MEL NO FRAMEWORK MIR_REF

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas de Informação do Campus de Três Lagoas da Universidade Federal de Mato Grosso do Sul como parte das exigências para a obtenção do título de Bacharel em Sistemas de Informação.

Professor orientador: Rodrigo Mitsuo Kishi

Três Lagoas/2025

Agradecimentos

Sou profundamente grato aos meus pais, Maria Lúcia e Dorival, por acreditarem em mim durante todos os momentos, por serem minha base, meu porto seguro e minha maior fonte de força. Sem o amor, o incentivo e o apoio de deles, este trabalho simplesmente não existiria. Agradeço também aos meus avós, Albertina e Armando, pelo carinho constante, pelas palavras de incentivo e por nunca terem duvidado de que eu seria capaz de chegar até aqui.

Também sou imensamente grato ao meu orientador, Rodrigo Mitsuo Kishi, por toda a sua paciência, disponibilidade e confiança ao longo não apenas deste trabalho, como toda a minha trajetória universitária. Suas orientações, sugestões e questionamentos foram fundamentais para que este trabalho tomasse forma e para que eu crescesse como estudante e como profissional.

Eu também só consegui chegar até aqui graças aos incríveis professores que tive ao longo do caminho: Rodrigo, Ivone, Ronaldo, Juliano, Humberto e Vitor. Com certeza, grande parte do do que sou hoje, tanto profissional quanto pessoal, devo a eles. Sou, de coração, eternamente grato.

Um agradecimento especial ao professor Vitor pela disponibilização dos recursos do laboratório.

E a todos os meus amigos que estiveram ao meu lado, mesmo que a quilômetros de distância, deixo minha mais sincera gratidão.

Resumo

Com o crescente consumo de conteúdo multimídia, a área de Recuperação da Informação Musical (MIR) tem se consolidado como um campo estratégico para a análise e interpretação de sinais sonoros. Nesse contexto, a extração de características de áudio, como o timbre, desempenha papel central em diversas tarefas, embora ainda enfrente desafios devido à complexidade e diversidade dos sinais musicais. Apesar do avanço das técnicas, ainda há uma lacuna quanto à comparação sistemática entre métodos tradicionais de extração de características e abordagens modernas baseadas em aprendizado profundo. Em especial, permanece o desafio de integrar e comparar diferentes tipos de representações de áudio em um mesmo ambiente experimental, de modo padronizado e reprodutível. Este trabalho tem como objetivo investigar a viabilidade da integração dos coeficientes cepstrais de frequência Mel (MFCC) ao *framework* `mir_ref`, possibilitando a inclusão dessa representação clássica em experimentos comparativos com métodos modernos, como *embeddings* gerados por redes neurais profundas. Além da inclusão dos MFCCs, avaliou-se a qualidade das representações extraídas por diferentes métodos sob um mesmo ambiente experimental, promovendo a reprodutibilidade e a padronização na avaliação de técnicas de MIR. Os experimentos realizados demonstram o impacto da escolha da representação nas tarefas de classificação musical, evidenciando as particularidades e limitações de cada abordagem. Assim, este estudo contribui para o entendimento comparativo entre técnicas tradicionais e modernas no contexto da representação de áudio.

Palavras-chave: Recuperação de Informação Musical. Processamento de áudio. `mir_ref`. MFCC. Representações de áudio.

Abstract

With the growing consumption of multimedia content, the field of Music Information Retrieval (MIR) has established itself as a strategic area for the analysis and interpretation of audio signals. In this context, the extraction of audio features, such as timbre, plays a central role in several tasks, but significant challenges remain due to the complexity and diversity of musical signals. Despite advances in the field, there is still a gap regarding the systematic comparison between traditional feature extraction methods and modern approaches based on deep learning. In particular, the challenge persists of integrating and comparing different types of audio representations within a standardized and reproducible experimental environment. This study aims to investigate the feasibility of integrating Mel-frequency cepstral coefficients (MFCCs) into the `mir_ref` framework, enabling the inclusion of this classical representation in comparative experiments with modern methods, such as embeddings generated by deep neural networks. In addition to the inclusion of MFCCs, the quality of the representations extracted by different methods was evaluated within the same experimental framework,

promoting reproducibility and standardization in MIR evaluation. The conducted experiments demonstrate the impact of representation choice on music classification tasks, highlighting the particularities and limitations of each approach. Therefore, this study contributes to a comparative understanding of traditional and modern techniques for audio representation.

Keywords: Music Information Retrieval. Audio processing. `mir_ref`. MFCC. Audio representations.

Lista de ilustrações

Lista de tabelas

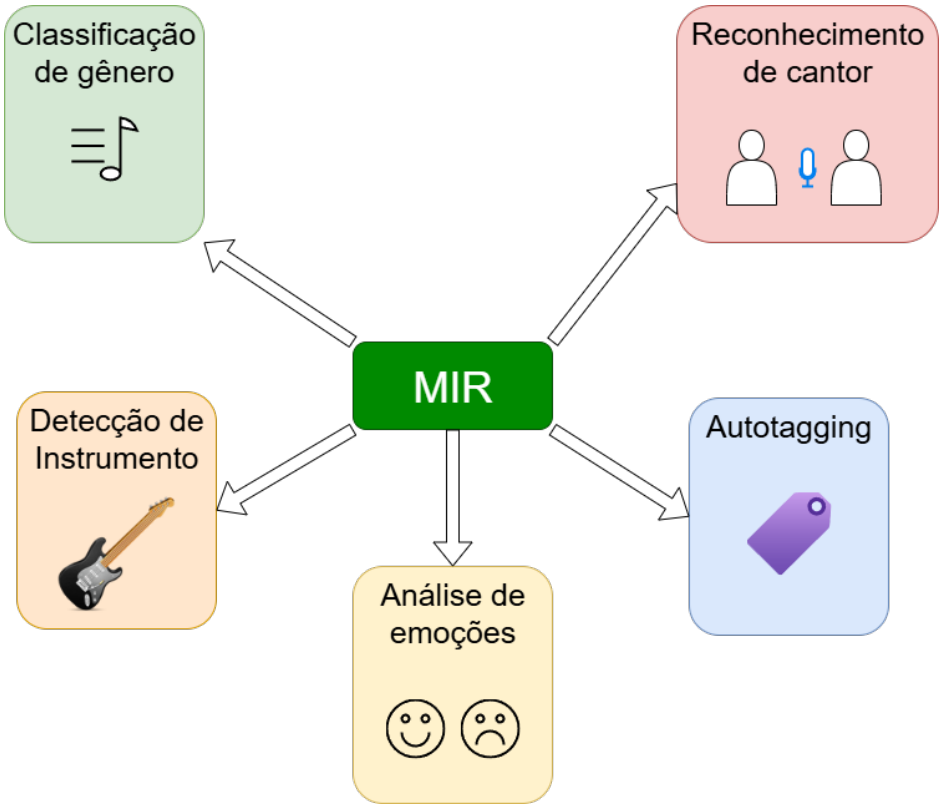
Sumário

1 Introdução

A Recuperação de Informação Musical (*Music Information Retrieval*, ou MIR) constitui uma área da ciência da computação dedicada à extração, processamento e interpretação de informações musicais a partir de sinais de áudio [1, 2]. Suas aplicações têm impacto direto em sistemas modernos de recomendação musical [1], assistentes virtuais [2], análise de tendências em redes sociais [3] e organização de grandes acervos sonoros [4].

Para que essas aplicações de MIR sejam viáveis, é necessário transformar o sinal de áudio, originalmente contínuo, em uma representação computacional que capture suas características mais relevantes. Essas representações podem ser projetadas manualmente, com base em conhecimento acústico, ou aprendidas automaticamente por modelos de aprendizado de máquina. A qualidade dessas representações influencia diretamente o desempenho em tarefas como classificação de gênero [1], detecção de instrumentos [5], análise de emoções [6], reconhecimento de cantor [7] e *autotagging* [1]. Um diagrama ilustrando algumas possibilidades de aplicações pode ser visualizado na Figura 1.

Figura 1 – Exemplos de aplicações da Recuperação da Informação Musical (MIR).



Podemos visualizar algumas tarefas fundamentais em MIR na Figura 1. No caso da classificação de gênero, o sistema precisa identificar a qual estilo musical, como rock, jazz ou música clássica, um determinado áudio pertence, com base nas suas características sonoras. Na detecção de instrumentos, busca-se identificar automaticamente quais instrumentos estão presentes no áudio analisado. Já a análise de emoções procura inferir sentimentos como tristeza, alegria ou tensão transmitidos pela música. O reconhecimento de cantor tem como

objetivo distinguir a voz dos intérpretes, permitindo, por exemplo, identificar quem está cantando em uma gravação. Por fim, o *autotagging* realiza a atribuição automática de palavras-chave (tags) que descrevem aspectos relevantes do conteúdo musical, facilitando a indexação e busca em grandes acervos.

Diversas ferramentas têm sido desenvolvidas com o objetivo de facilitar tanto a extração de características quanto a avaliação dessas representações. Entre elas, destacam-se bibliotecas como o *mirdata* [4], que padroniza o acesso a conjuntos de dados, e o *mir_eval* [8], responsável por fornecer medidas de avaliação comuns para tarefas da área.

Com o desenvolvimento da aprendizagem profunda, diferentes arquiteturas de redes neurais vêm sendo exploradas para gerar representações a partir dos dados. Um dos formatos mais utilizados como entrada para esses modelos é o espectrograma, uma representação bidimensional que expressa a distribuição da energia sonora ao longo do tempo e das frequências. Essa estrutura é especialmente adequada para o uso de redes convolucionais, como na arquitetura *MusicNN*, que aprende *embeddings* diretamente a partir de espectrogramas [9].

Embeddings são vetores numéricos de baixa dimensionalidade que representam dados complexos, como sinais de áudio, em um espaço contínuo [10]. No contexto de MIR, esses vetores são gerados por modelos treinados em grandes coleções musicais, codificando informações como padrões temporais, frequências dominantes e relações harmônicas [11, 9, 12]. Seu uso tem se popularizado por permitir representações compactas, eficientes e comparáveis, sendo amplamente empregadas em tarefas de classificação, recuperação e análise de similaridade musical [13, 7, 14].

Além das representações espectrotemporais, outras abordagens propõem o uso de fontes de informação complementares, como metadados musicais [15], relações semânticas em playlists [16] e dados visuais extraídos de vídeos musicais [13].

Entretanto, quando representações e ferramentas do ecossistema são adotadas de forma isolada, com pré-processamentos, formatos de entrada e configurações experimentais distintos, a reprodutibilidade dos estudos fica dificultada, isto é, torna-se mais difícil replicar resultados sob os mesmos dados, parâmetros e procedimentos [17, 14].

Para enfrentar esses desafios, algumas iniciativas recentes buscaram aprimorar a padronização e a reprodutibilidade na área de Recuperação da Informação Musical. O MIREX (*Music Information Retrieval Evaluation eXchange*) organiza avaliações comunitárias anuais, oferecendo medidas e conjuntos de dados padronizados para comparação de sistemas [18]. O MARBLE (*Music Audio Representation Benchmark for Universal Evaluation*) propõe uma taxonomia abrangente e um protocolo unificado para avaliação de representações musicais [14]. Já o MIRFLEX (*Music Information Retrieval Feature Library for Extraction*) concentra-se na modularização de modelos de extração de características como timbre, tonalidade e ritmo, proporcionando um sistema flexível para análise musical [19].

Essas iniciativas ilustram a diversidade de estratégias disponíveis na literatura e reforçam a importância de dispor de um ambiente unificado em que diferentes representações

possam ser comparadas sob as mesmas tarefas, medidas e protocolos de treino. Entre as ferramentas que buscam oferecer esse tipo de ambiente integrado, destaca-se a biblioteca `mir_ref`, proposta por Plachouras *et al.* [20]. O `mir_ref` é um *framework* voltado à avaliação de representações de áudio que centraliza e padroniza, em um único *pipeline*, o acesso a diferentes conjuntos de dados, a extração de representações de áudio, o treinamento de classificadores sob protocolos padronizados e medidas de avaliação.

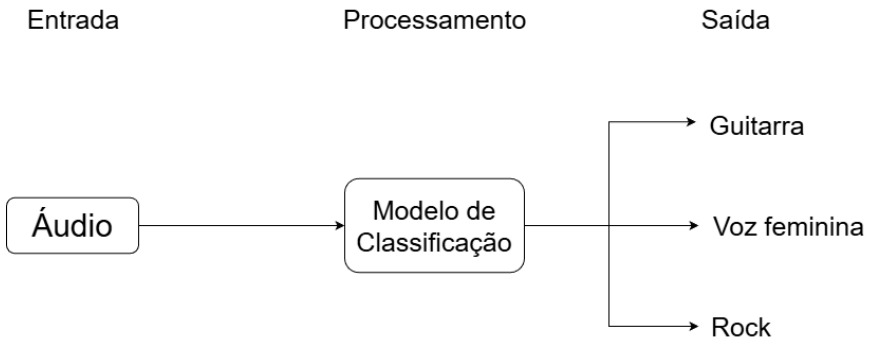
Considerando essa flexibilidade do `mir_ref` e o fato de que, até o momento, o *framework* não inclui uma implementação nativa dos coeficientes cepstrais de frequência Mel (MFCC – *Mel-Frequency Cepstral Coefficients*), este trabalho tem como objetivo principal estender o `mir_ref` com um módulo de extração de MFCCs totalmente integrado ao seu *pipeline*. Em outras palavras, a contribuição central não é apenas estudar os MFCCs em si, mas disponibilizá-los como uma opção de representação clássica, reprodutível e configurável dentro do mesmo ambiente em que já se encontram as representações neurais modernas.

Os MFCCs destacam-se por sua simplicidade, baixo custo computacional e alta capacidade de capturar informações relevantes sobre a estrutura timbrística do som, sendo amplamente utilizados tanto em reconhecimento de fala quanto em classificação musical. Além disso, continuam a ser uma das principais escolhas como *baseline* em experimentos comparativos com abordagens mais recentes baseadas em aprendizado profundo [21, 11, 12].

Essa integração será feita de modo que ela seja compatível com qualquer uma das tarefas suportadas pela plataforma. Para fins de validação do sucesso dessa integração, foram conduzidos experimentos na tarefa de *autotagging* musical, tratada como um problema de classificação multirrótulo, em que cada áudio pode estar associado a múltiplos rótulos simultaneamente.

Em classificação multirrótulo um único trecho de áudio pode ser descrito simultaneamente por diferentes categorias, como instrumentos, características vocais e estilos. Esse cenário exige que o modelo reconheça múltiplos padrões acústicos de forma conjunta e pode ser melhor visualizado na Figura 2.

Figura 2 – Exemplo de classificação multirrótulo no contexto de *autotagging* musical. Cada áudio pode apresentar múltiplos rótulos simultaneamente, como instrumentos, estilos musicais e características vocais.



Organização do texto. A Seção 2 apresenta os conceitos fundamentais do trabalho:

digitalização de áudio, extração de características, métodos de classificação de áudio, bases de dados e medidas de avaliação. A Seção 3 descreve a proposta central deste estudo, isto é, a metodologia de integração dos MFCCs ao `mir_ref`, com foco na extração, na parametrização e no encaixe dessa representação no fluxo do *pipeline*. A Seção 4 mostra a aplicação da nova extensão em um estudo de caso, apresentando o protocolo de treinamento, os classificadores padronizados e os resultados da comparação entre MFCC, CLMR e VGGish, com o objetivo de validar essa integração. A Seção 5 discute implicações e limitações, aponta lacunas de pesquisa e enumera os trabalhos futuros.

2 Conceitos Relacionados

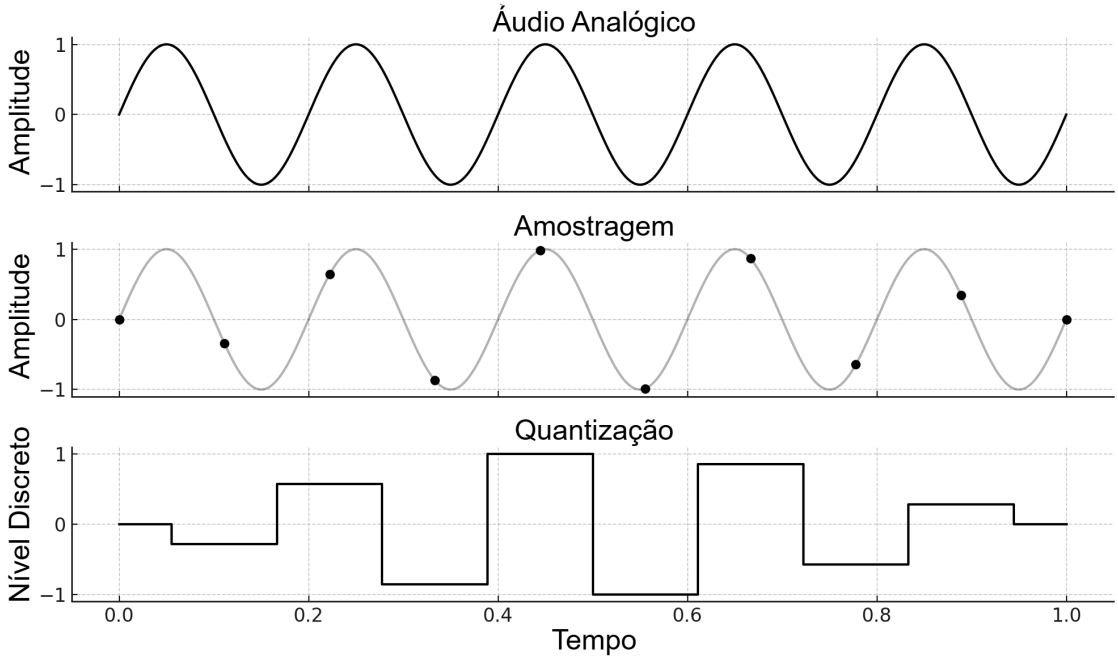
Esta seção apresenta os principais conceitos que fundamentam este trabalho. São introduzidos os temas de representação digital de sinais sonoros, abordagens de extração de características, métodos para classificação de áudio, bases de dados especializadas, medidas de desempenho e protocolos de comparação.

2.1 Áudio Digital

O som é, em sua essência, uma onda mecânica, gerada pela vibração de partículas em meios físicos como o ar, a água ou materiais sólidos [22]. Essas vibrações produzem variações de pressão que se propagam pelo ambiente e podem ser captadas tanto pelo ouvido humano quanto por dispositivos eletrônicos. A descrição científica desse fenômeno considera parâmetros como frequência, amplitude, fase e velocidade de propagação [22, 23, 24].

Para possibilitar seu processamento e armazenamento digital, é necessário converter o sinal sonoro, originalmente contínuo, em uma sequência de valores discretos. Essa digitalização envolve dois aspectos principais: a amostragem e a quantização. A amostragem corresponde ao registro do valor do sinal em intervalos regulares de tempo, enquanto a quantização associa cada uma dessas amostras a um nível discreto, definido em bits.

Figura 3 – Processo de digitalização de áudio, destacando as etapas de amostragem (discretização no tempo) e quantização (discretização em amplitude).



Esse processo de digitalização de áudio é ilustrado na Figura 3. O gráfico superior representa um sinal analógico contínuo, como uma onda sonora real captada por um microfone. No gráfico central, observamos o processo de amostragem: os pontos pretos indicam os instantes em que o sinal é medido, com base em uma taxa de amostragem constante. Por fim, o gráfico inferior mostra a quantização, na qual cada valor amostrado é associado ao nível discreto mais próximo disponível. Essa transformação em níveis fixos de amplitude é o que permite representar o sinal em formato digital.

A qualidade da conversão depende diretamente da taxa de amostragem e da quantidade de bits, também conhecida como profundidade de bits. A taxa de amostragem determina quantas amostras são coletadas por segundo e, consequentemente, qual faixa de frequências pode ser representada de maneira precisa. A profundidade de bits, por sua vez, define o número de níveis distintos disponíveis para representar cada amostra, influenciando diretamente a resolução dinâmica do sinal. Por exemplo, uma profundidade de 16 bits permite 65.536 níveis de intensidade sonora, garantindo maior fidelidade do que uma profundidade de 8 bits, que permite 256 níveis na preservação de detalhes do sinal original [25].

2.2 Extração de Características

A extração de características é uma etapa essencial no processamento de sinais de áudio, pois permite transformar o sinal de áudio digitalizado, isto é, a sequência de amostras de amplitude ao longo do tempo, em representações mais compactas, informativas e adequadas para análise automática [2, 26]. Essa transformação visa reduzir a complexidade

dos dados, mantendo os aspectos mais relevantes para tarefas específicas, como classificação, reconhecimento de padrões ou segmentação.

Diferentes tipos de características podem ser extraídas, cada uma delas fornecendo informações distintas sobre o sinal. As principais categorias são:

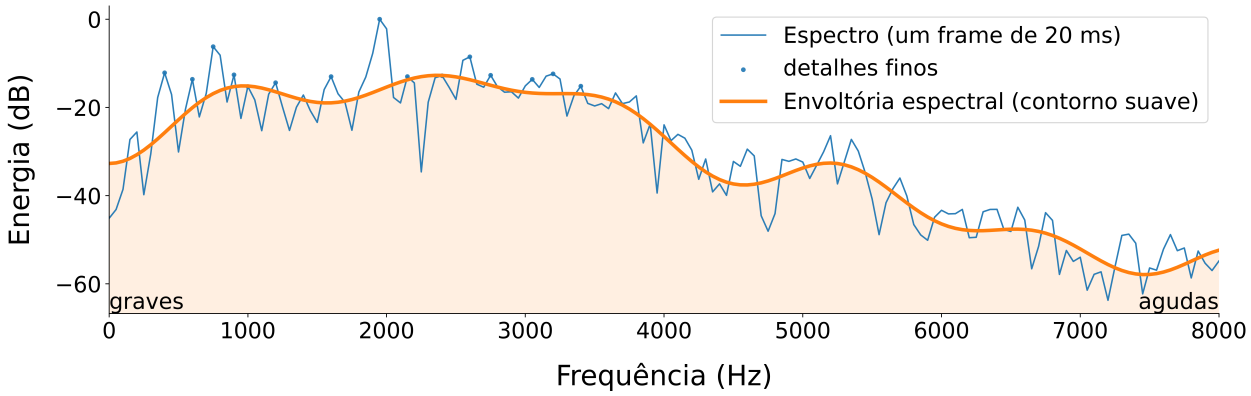
- **Temporais:** descrevem como o sinal varia ao longo do tempo. Exemplos incluem o *zero-crossing rate*, que calcula a taxa de cruzamentos por zero do sinal, útil para distinguir entre sons harmônicos e ruidosos [27, 1];
- **Espectrais:** analisam a distribuição de energia em diferentes faixas de frequência. Entre os principais exemplos estão o *spectral centroid*, que indica onde a energia do espectro está concentrada e está associado à percepção de brilho do som [27]; o *spectral bandwidth*, que mede a dispersão das frequências ao redor do centróide [27]; e o *spectral roll-off*, que representa a frequência abaixo da qual se encontra uma porcentagem acumulada da energia total do espectro [27].
- **Harmônicas:** exploram a organização tonal do sinal. As *chroma features* representam a distribuição de energia entre as 12 classes de notas musicais dentro de uma oitava [2, 1]. Outros exemplos incluem o *harmonic-to-noise ratio* (HNR), que quantifica o equilíbrio entre componentes harmônicos e ruidosos, e a *tonnetz* (ou *tonal centroid features*), que capturam relações harmônicas baseadas na teoria musical [2].
- **Perceptuais:** modelam a percepção auditiva humana. O exemplo clássico são os MFCCs (*Mel-Frequency Cepstral Coefficients*), amplamente utilizados em reconhecimento e classificação musical [21, 27, 1]. Variantes incluem os GFCCs (*Gammatone Frequency Cepstral Coefficients*), baseados em filtros *gammatone* [28], e coeficientes em escala de Bark, que organizam energia por bandas críticas [29].
- **Representações neurais:** vetores gerados por redes neurais profundas treinadas em grandes coleções de áudio. Diferem das categorias anteriores por se definirem sobretudo pelo processo de obtenção, o modelo aprende a representar o áudio de forma compacta, em vez de por um único aspecto do conteúdo do sinal; por isso, tendem a integrar, de forma conjunta, pistas temporais, espectrais e harmônicas. Exemplos incluem CLMR, aprendizado contrastivo auto-supervisionado [12]; VGGish, modelo genérico treinado em um acervo amplo de sons do cotidiano [11]; OpenL3, representação multimodal que aprende a alinhar áudio e imagem em vídeos [13]; e MusicNN, modelo voltado especificamente para música, focado em padrões timbrais e rítmicos usados em *autotagging* [9].

2.3 Coeficientes Cepstrais de Frequência Mel (MFCC)

No procedimento habitual, o áudio é segmentado em pequenos trechos sucessivos de 20 a 40 ms, isto é, janelas curtas, também chamadas *frames*. Os *Mel-Frequency Cepstral*

Coefficients, MFCCs, formam um conjunto de números que, para cada *frame*, descreve o contorno geral de energia entre frequências graves e agudas, correspondendo à envoltória espectral de curto prazo. Em vez de acompanhar os detalhes finos do espectro, observa-se esse contorno superior, que resume quais faixas de frequência estão mais fortes, de modo análogo ao visor de um equalizador [25, 2]. A Figura 4 ilustra a ideia: em azul, o espectro em decibéis de um *frame* de 20 ms com sua estrutura fina, marcada por picos espectrais; em laranja, a envoltória, traço suave que resume a distribuição de energia dos graves para os agudos. As medições são realizadas na frequência Mel, que organiza a frequência de modo alinhado à audição humana, com maior resolução perceptual nas regiões graves do que nas muito agudas [29, 21, 27]. Neste trabalho, os MFCCs são utilizados como indicador de timbre, isto é, um resumo numérico desse contorno espectral ao longo do tempo, útil em tarefas de classificação e recuperação musical [1, 21, 27].

Figura 4 – Espectro em dB de um *frame* de 20 ms em azul e sua envoltória espectral em laranja. Os pontos azuis assinalam picos espectrais. Escalas em dB normalizadas para visualização.



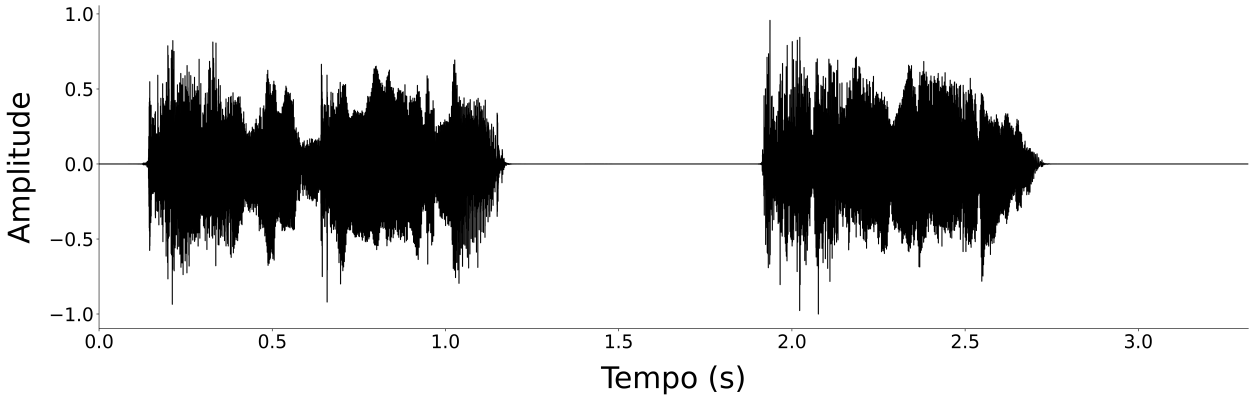
Neste trabalho, o *timbre* é adotado no sentido clássico descrito na literatura de psicoacústica e de MIR: a característica perceptual que distingue sons que têm a mesma altura, a mesma intensidade e a mesma duração. Nesse sentido, o timbre envolve o contorno médio do espectro em janelas curtas, ou seja, a envoltória espectral de curto prazo, e traços temporais como ataque e decaimento, além de componentes ruidosos e modulações que moldam a impressão auditiva do som [29, 2].

Para cada *frame*, o espectro de potência é agregado por um banco de filtros na frequência Mel, resultando em um vetor de energias por banda. Em seguida, aplica-se uma compressão logarítmica e a Transformada Discreta do Cosseno do tipo II, com normalização ortonormal. Essa transformação cumpre dois papéis. Primeiro, reduz redundâncias: como as energias das bandas Mel costumam variar juntas, o novo conjunto de coeficientes tende a variar de modo mais independente. Segundo, organiza a informação por grau de suavidade: os coeficientes de índice menor descrevem a forma geral da envoltória, enquanto os de índice maior registram variações rápidas entre bandas [25, 27].

2.3.1 Sinal no tempo

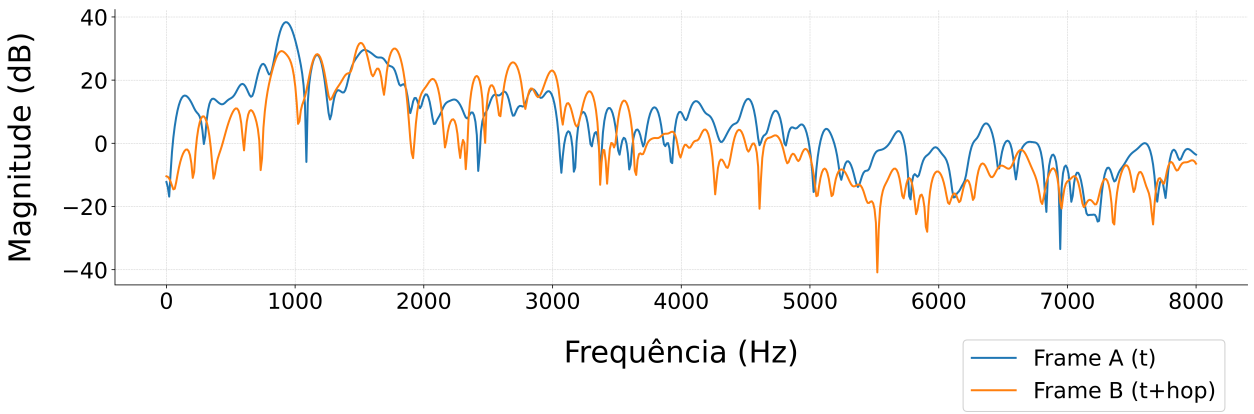
A forma de onda é a amplitude ao longo do tempo, e um exemplo para um determinado áudio pode ser visualizado na Figura 5. Em *frames* de 20 a 40 ms, a estrutura espectral tende a mudar pouco dentro de cada janela; por isso cada trecho é tratado como um "instantâneo" do som, no regime de quase-estacionariedade, e as medições desses instantâneos são alinhadas ao longo do tempo [25, 2].

Figura 5 – Forma de onda no domínio do tempo, com regiões ativas de alta amplitude e um intervalo de silêncio visível.



Dois *frames* adjacentes em um trecho quase estacionário, com duração de 20 ms e 50% de sobreposição, quando apresentam espectros que praticamente se sobrepõem, indicam que a distribuição de energia por frequência permanece quase a mesma de um *frame* para o seguinte, típica da sustentação de sons como uma vogal prolongada, uma nota mantida ou um ruído constante, como ilustrado na Figura 6.

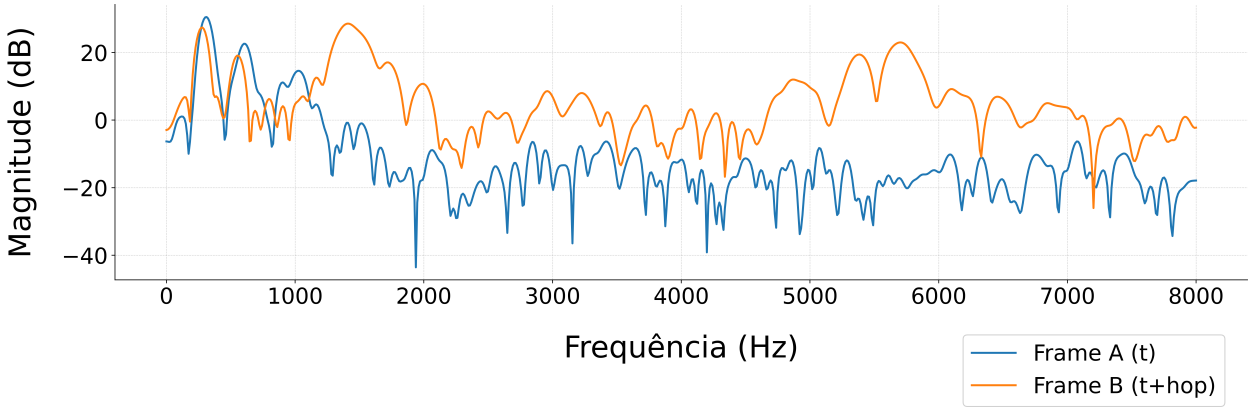
Figura 6 – Espectro de potência em decibéis de dois *frames* adjacentes com duração de 20 ms e sobreposição de 50%. A semelhança entre as curvas evidencia a quase-estacionariedade em janelas curtas. Eixo de frequência limitado a 8 kHz.



Em ataques e transientes, como inícios de notas, batidas e consoantes explosivas, a energia cresce e se redistribui rapidamente, sobretudo nas altas frequências: picos e vales do espectro mudam de posição e as curvas se afastam, evidenciando a mudança. Na Figura 7,

essa situação é exemplificada por dois *frames* adjacentes em um transiente, cuja diferença local sinaliza variação temporal rápida, o tipo de mudança que a análise por janelas curtas procura isolar [25, 2].

Figura 7 – Espectro de potência em decibéis de dois *frames* adjacentes em região de transiente, com duração de 20 ms e sobreposição de 50%. A diferença entre as curvas reflete a redistribuição rápida de energia ao longo das frequências, típica de ataques e outros eventos não estacionários.



Formalmente, um *frame* é um trecho contíguo do sinal com N_w amostras, correspondente ao tamanho da janela. O avanço H define o início do *frame* seguinte, e a sobreposição entre janelas é dada por $(N_w - H)/N_w$. Dadas a taxa de amostragem `sr` e as durações desejadas τ_w para a janela e τ_h para o avanço, os valores inteiros usados no processamento são

$$N_w \;=\; \mathrm{round}\big(\tau_w \,\mathrm{sr}\big), \qquad H \;=\; \mathrm{round}\big(\tau_h \,\mathrm{sr}\big).$$

Essas durações no tempo

$$\tau_w \;=\; \frac{N_w}{\mathrm{sr}}, \qquad \tau_h \;=\; \frac{H}{\mathrm{sr}}$$

servem para expressar os tamanhos em milissegundos, posicionar cada *frame* no eixo temporal, e interpretar a sobreposição ao longo do sinal. Considerando um sinal com comprimento L em amostras, o número de *frames* obtidos ao deslizar a janela de N_w amostras com avanço H é

$$T \;=\; \left\lfloor \tfrac{L-N_w}{H} \right\rfloor + 1 \quad (L \geq N_w),$$

isto é, contamos todas as posições em que a janela cabe inteiramente dentro do sinal.

Exemplo numérico. Considere sr = 22,05 kHz, janela de 20 ms e avanço de 10 ms.

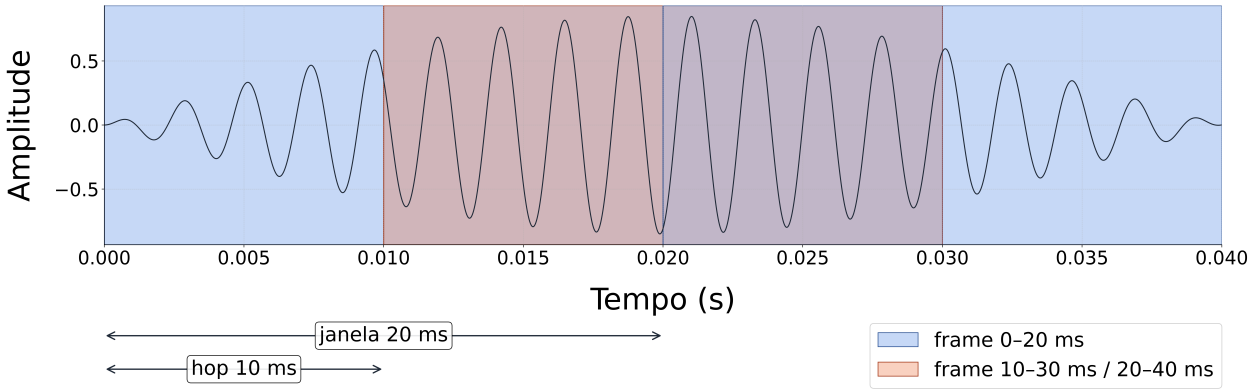
$$N_w = \mathrm{round}(0{,}020 \cdot 22050) = 441, \qquad H = \mathrm{round}(0{,}010 \cdot 22050) = 220.$$

A fração de sobreposição é $(N_w - H)/N_w = 221/441 \approx 0{,}501$ — cerca de 50%. Para um trecho de 1 segundo com $L = 22050$ amostras, o número de *frames* é

$$T = \left\lfloor \tfrac{L-N_w}{H} \right\rfloor + 1 = \left\lfloor \tfrac{22050-441}{220} \right\rfloor + 1 = 99.$$

Em tempo contínuo, o *frame* 0 cobre 0–20 ms, o *frame* 1 cobre 10–30 ms e o *frame* 2 cobre 20–40 ms, deixando explícita a sobreposição de metade da janela. A Figura 8 mostra esse início do sinal: cada faixa azul corresponde a uma janela de 20 ms; a região de cor mesclada evidencia a sobreposição de 50%; as setas indicam a largura da janela e o avanço, também chamado de *hop*, de 10 ms.

Figura 8 – Início do sinal com três *frames* consecutivos. Cada faixa azul marca uma janela de 20 ms. A região de cor mesclada evidencia a sobreposição de 50%. As setas indicam a largura da janela e o avanço de 10 ms.



O tamanho da janela N_w regula o equilíbrio entre resolução temporal e resolução em frequência. Janelas mais longas estreitam os picos no espectro e ajudam a distinguir frequências próximas; um exemplo é a separação entre as notas musicáis $\text{LÁ}_4 \approx 440$ Hz e $\text{LÁ}\sharp_4 \approx 466$ Hz, diferença de cerca de 26 Hz. Janelas muito curtas alargam esses picos e podem fundir tons vizinhos em um único máximo. Em compensação, janelas curtas preservam melhor eventos muito próximos no tempo, como duas palmas separadas por poucos milissegundos ou um ataque de caixa seguido de um prato, e capturam microvariações temporais como modulações de altura e amplitude [25, 2].

O avanço entre *frames* H estabelece a amostragem temporal e a fração de sobreposição $(N_w - H)/N_w$. Avanços menores produzem sequências mais densas de *frames* e transições mais suaves entre *frames* sucessivos [1, 27]. Em contrapartida, avanços maiores reduzem a sobreposição e rarefazem a amostragem temporal; eventos de curta duração ou variações rápidas na envoltória podem cair entre *frames*, e a leitura temporal torna-se menos suave [25, 2]. Com uma função de janela do tipo Hann, aplicada a cada *frame*, na forma periódica adotada nesta análise, é comum usar $H \approx N_w/2$, o que gera sobreposição em torno de 50%. Essa configuração satisfaz a condição de *constant overlap-add* (COLA) [30, 31]: a soma ponto a ponto de $w[n]$ e $w[n + H]$ mantém-se aproximadamente constante nas regiões de sobreposição. O efeito prático é uma estimativa espectral mais estável ao longo do tempo: o sinal não oscila por causa do janelamento e da sobreposição, e a envoltória espectral é lida com mais consistência [25, 30].

Antes da análise em frequência, cada *frame* é multiplicado por uma função de ponderação, também chamada função de janela. Essa função é uma sequência de pesos $w[n]$

aplicada ponto a ponto ao trecho $x_m[n]$. Cada peso atua como ganho: valores próximos de 0 enfraquecem a amostra correspondente e valores próximos de 1 a preservam. O sinal janelado é

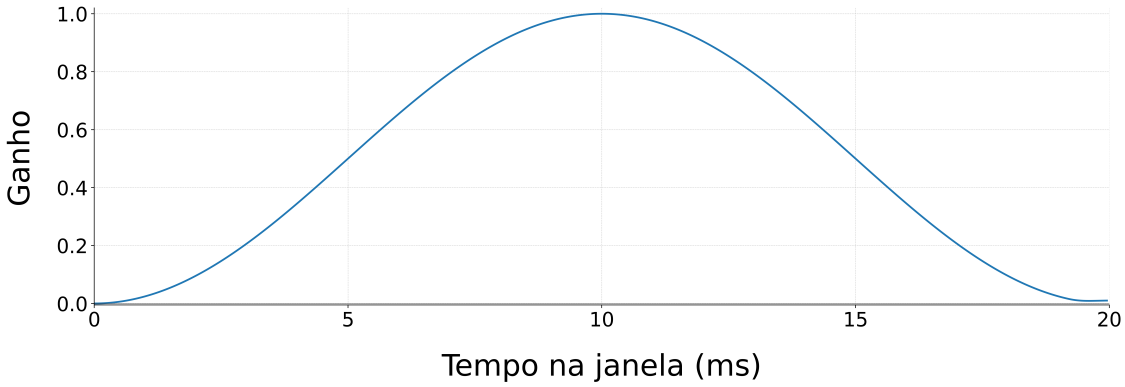
$$x_m^{(w)}[n] \;=\; x_m[n]\, w[n].$$

A função de janela de Hann, na forma periódica adotada na STFT, é aplicada ao longo de cada *frame* de 20 ms para suavizar o início e o fim do trecho analisado. Como mostrado na Figura 9, o eixo horizontal representa o tempo dentro do *frame* e o eixo vertical, o ganho aplicado a cada amostra. Na Hann periódica, o ganho é máximo no centro do *frame*, igual a 1, e decai suavemente: é zero no início do *frame* e permanece muito próximo de zero no fim. A expressão utilizada é

$$w[n] \;=\; \tfrac{1}{2} \;-\; \tfrac{1}{2}\cos\!\left(\tfrac{2\pi n}{N_w}\right), \qquad n = 0, \ldots, N_w - 1,$$

com N_w amostras na janela [25, 27].

Figura 9 – Função de janela de Hann ao longo do *frame* de 20 ms na forma periódica usada na STFT. Eixo x: tempo na janela em milissegundos. Eixo y: ganho multiplicativo aplicado a cada amostra. A curva inicia em zero e, no final do *frame*, permanece ligeiramente acima de zero, o que reduz vazamento espectral.



A *Short-Time Fourier Transform* (STFT) supõe que cada *frame* se repete infinitamente para a esquerda e para a direita. Quando o corte é abrupto, a última e a primeira amostras não coincidem e surge uma descontinuidade. Essa aresta cria borrões no espectro: aparece energia onde não há conteúdo espectral do sinal e parte da energia dos picos vaza para frequências vizinhas. Esse vazamento reduz o contraste entre picos e dificulta mensurar a contribuição de cada pico espectral. A função de janela de Hann atenua as bordas e reduz esse vazamento, tornando a estimativa mais estável de um *frame* para o seguinte. A contrapartida é um pico principal um pouco mais largo no domínio da frequência, isto é, resolução menor entre picos muito próximos. Ainda assim, o objetivo aqui é captar o contorno geral de energia com estabilidade, não separar picos quase coincidentes, de modo que esse pequeno alargamento não compromete a análise [25, 2].

Voltando ao exemplo com $N_w = 441$ e avanço $H = 220$, o que da um *hop* de aproximadamente 10 ms, os três *frames* iniciais cobrem 0–20 ms, 10–30 ms e 20–40 ms. A

função janela de Hann reduz a descontinuidade entre *frames* ao atenuar a transição entre o final de um *frame* e o inicio do seguinte. Com 50% de sobreposição, essas bordas se encaixam e a soma ponto a ponto mantém-se praticamente constante ao longo do tempo [25, 27].

2.3.2 STFT e espectro de potência

A Transformada Discreta de Fourier (DFT - *Discrete Fourier Transform*) transforma um trecho janelado do sinal em um conjunto de coeficientes espectrais igualmente espaçados em frequência. A magnitude de cada coeficiente indica quanta energia há na frequência correspondente dentro do trecho analisado. Como a DFT assume que o bloco de N amostras é copiado e repetido sem fim, unindo a última à primeira amostra, o resultado resume o conteúdo de frequências do intervalo inteiro e não localiza eventos no tempo. Apesar dessa limitação temporal, passamos ao domínio da frequência porque é nele que o timbre se organiza, por meio do contorno espectral de curto prazo. Para captá-la, usamos o espectro de potência, uma estimativa estável da energia por frequência obtida a partir do quadrado do módulo dos coeficientes [25, 2, 21, 27].

Do ponto de vista computacional, quem torna esse cálculo viável janela a janela é a Transformada Rápida de Fourier (FFT - *Fast Fourier Transform*). Trata-se de uma família de algoritmos que calcula a DFT com complexidade de tempo $N \log N$, em vez de N^2. Essa economia permite repetir a DFT em cada *frame* ao longo de todo o sinal. A sequência dessas DFTs constitui a STFT e sustenta, na prática, a análise tempo–frequência usada aqui [25, 2].

Sinais de música e fala variam no tempo; uma DFT mistura eventos de instantes diferentes e perde a sequência temporal. A Transformada de Fourier de Tempo Curto (STFT - *Short-Time Fourier Transform*) resolve esse problema ao aplicar janelas de curta duração ao sinal e calcular a DFT de cada janela deslocada no tempo. O resultado é uma sequência de espectros indexados por *frame*, isto é, um espectrograma que mostra quando e em quais frequências a energia aparece. Por preservar a evolução temporal sem abrir mão da estimativa em frequência, a STFT é preferida para áudio não estacionário [25, 31, 2].

Para cada *frame* m, calcula-se a STFT

$$X[m,k] \;=\; \sum_{n=0}^{N_w-1} x[n+mH]\; w[n]\; e^{-j\,2\pi kn/N_{\mathrm{FFT}}}.$$

Nessa expressão, n varia de 0 a $N_w - 1$ e indica a posição temporal dentro do *frame*; k percorre a régua de frequências definida por N_{FFT} e corresponde à frequência $f_k = k\,\mathrm{sr}/N_{\mathrm{FFT}}$. O termo $x[n+mH]$ seleciona o trecho do sinal que começa em mH; a janela $w[n]$ suaviza as bordas desse trecho e reduz descontinuidades; o fator exponencial $e^{-j\,2\pi kn/N_{\mathrm{FFT}}}$ atua como um sintonizador centrado em f_k. A soma agrega a contribuição de todas as amostras para essa frequência. O coeficiente $X[m,k]$ é complexo: sua magnitude quantifica a energia em torno de f_k no *frame* e sua fase indica o alinhamento desse componente. [25, 2]

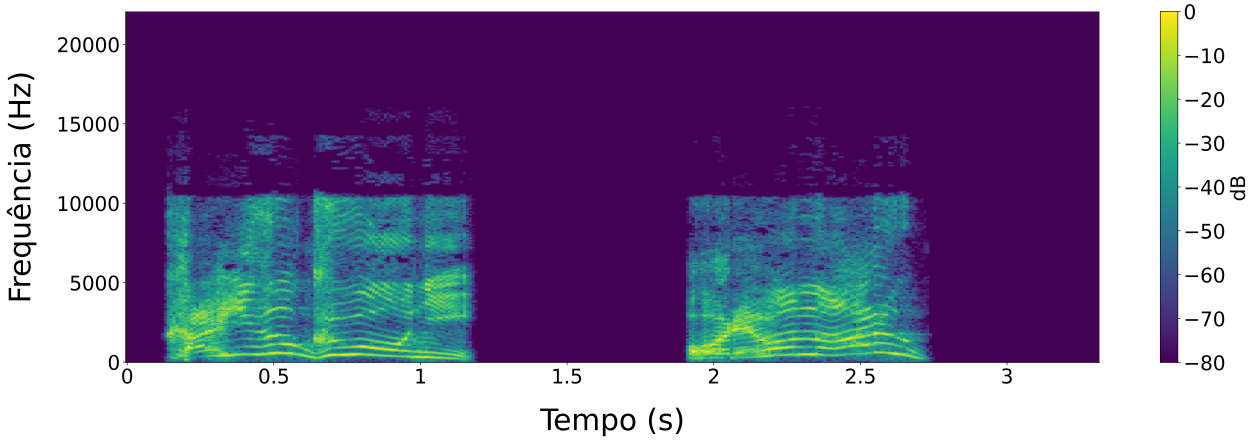
O parâmetro N_{FFT} define o tamanho da transformada aplicada ao *frame* e, com isso, fixa a régua de frequências onde a energia é medida. Aumentar N_{FFT} torna essa régua mais

densa e reduz o passo $\Delta f = \mathrm{sr}/N_{\mathrm{FFT}}$. Quando $N_{\mathrm{FFT}} > N_w$, completa-se o trecho com zeros apenas para avaliar a mesma curva espectral em mais pontos. Esse procedimento facilita a leitura e a integração por bandas, mas não cria informação nova nem aumenta a capacidade de separar frequências muito próximas, que continua determinada pela duração e pela forma da janela $w[n]$. [25, 2]

A grandeza utilizada adiante é o espectro de potência $|X[m,k]|^2$, medida estável de energia por frequência adequada à etapa seguinte de agregação nas bandas de frequência Mel. Ao empilhar $|X[m,k]|^2$ ao longo de m, obtém-se o espectrograma, que revela como a distribuição de energia em frequência evolui no tempo. [25, 31, 2]

Espectrogramas de potência mostram $|X[m,k]|^2$ em decibéis ao longo do tempo e da frequência em que cores mais claras indicam maior energia, isso pode ser melhor visualizado na Figura 10. Linhas quase horizontais repetidas em $f_0, 2f_0, 3f_0, \ldots$ são os harmônicos. O f_0 é a frequência fundamental, ou seja, a taxa básica de vibração de uma fonte periódica, e corresponde à altura percebida. Seus múltiplos inteiros formam a série harmônica que compõem o timbre e aparecem como linhas igualmente espaçadas verticalmente no espectrograma. Piano, violão ou uma vogal sustentada são exemplos que exibem esse padrão de linhas paralelas. Quando a altura sobe ou desce, todas as linhas se deslocam em conjunto e se inclinam; ondulações lentas e regulares nessas linhas revelam *vibrato*. Colunas claras e estreitas marcam *onsets*, ou ataques, como a palhetada inicial de uma corda, o toque de uma tecla ou o golpe de baqueta. Trechos com energia irregular e espalhada por muitas frequências formam texturas ruidosas, típico do som de pratos de bateria, ruído de vento e das consoantes "s" e "sh". Após um ataque, o escurecimento gradual indica o *decaimento*: a energia diminui enquanto a nota se dissipa até o silêncio. Esses padrões — harmônicos, ataques, ruído e silêncio — formam a leitura visual que as etapas seguintes condensam: primeiro pela soma em bandas na frequência Mel e, depois, pelos MFCCs, que resumem o contorno espectral [2].

Figura 10 – Espectrograma STFT em dB. Escala relativa ao máximo do trecho: 0 dB no valor máximo e −80 dB como piso dinâmico. Tons claros indicam maior energia e tons escuros aproximam-se do silêncio. Harmônicos surgem como faixas quase horizontais, ataques como colunas claras e regiões ruidosas como manchas largas.



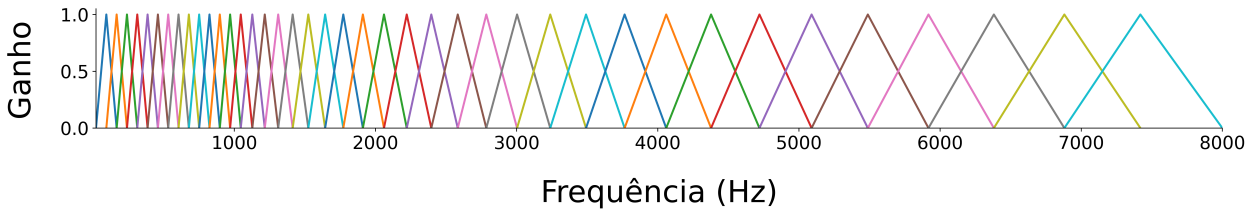
2.3.3 Projeção em banco de filtros Mel

Depois de estimar o espectro de potência em cada *frame*, reunimos essa energia em bandas perceptuais. O banco de filtros Mel faz a passagem do eixo de frequência para um eixo alinhado à percepção humana. Nas frequências baixas a audição tem maior resolução e distingue intervalos menores de frequência do que nas altas. Por isso o banco coloca filtros mais densos no grave e mais espaçados no agudo [21, 27].

A projeção é uma soma ponderada das energias do espectro por filtros triangulares definidos na frequência Mel. Cada filtro cobre uma faixa de frequências, atribui pesos que crescem até a frequência central e decrescem até zero nas bordas. Em cada *frame*, multiplicamos a potência espectral pelos pesos do filtro e somamos, repetindo o procedimento para todas as bandas para formar um vetor de energias Mel. Esse vetor reduz a dimensionalidade, suaviza ondulações muito finas, e concentra a informação mais relevante para o timbre [21, 27].

No painel do banco de filtros, ver Figura 11, mostramos o banco espaçado uniformemente na frequência Mel, representado aqui no eixo de frequência em hertz. Cada triângulo atua como função de ganho com valor 1 no centro e 0 nas bordas; o vértice central marca a frequência central da banda. Filtros vizinhos se sobrepõem de forma contínua, evitando lacunas na agregação de energia entre bandas. Observa-se que, no grave, os filtros ficam mais próximos; no agudo, as bases se alargam em hertz, acompanhando a menor resolução auditiva nessa região.

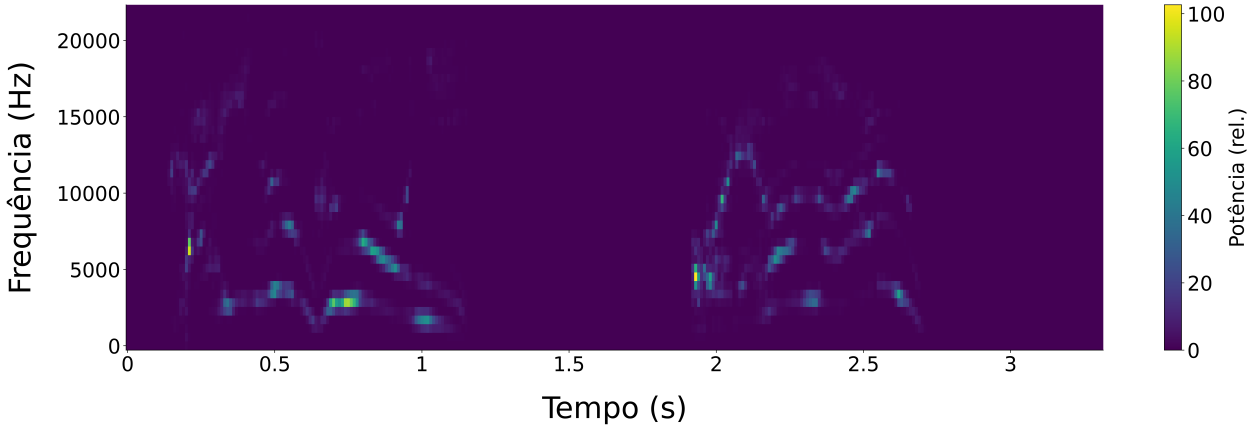
Figura 11 – Banco de filtros Mel representado em hertz. Cada triângulo tem ganho 1 no centro e 0 nas bordas; os vértices centrais indicam as frequências centrais. O espaçamento uniforme em Mel gera maior densidade no grave e bases mais largas no agudo.



O mapa de energias por banda ao longo do tempo, mostrado na Figura 12, empilha no tempo os vetores de energias Mel produzidos em cada *frame*. O resultado é uma matriz: as linhas correspondem às bandas Mel e as colunas aos *frames*. Assim, cada coluna é um instantâneo da distribuição de energia por banda naquele momento, e a varredura horizontal revela a evolução temporal. O eixo horizontal representa o tempo do sinal e o vertical lista as frequências centrais em hertz. As cores representam a potência em escala linear, normalizada ao máximo do trecho, a barra "Potência (rel.)" vai de 0 a 100, onde 100 é o maior valor observado e 0 é o mínimo, com tons mais claros indicando energias maiores. Faixas quase horizontais revelam componentes estáveis, trilhas inclinadas marcam variações de altura, realces nas bandas altas indicam textura ruidosa e regiões escuras sinalizam silêncio ou níveis

muito baixos de energia. Essa representação já organiza o espectro em unidades perceptuais e prepara a passagem para a compressão logarítmica e para a DCT.

Figura 12 – Energia por banda Mel ao longo do tempo em escala linear, normalizada ao máximo do trecho. Cores claras indicam maior energia; o eixo vertical indica as frequências centrais das bandas em hertz.

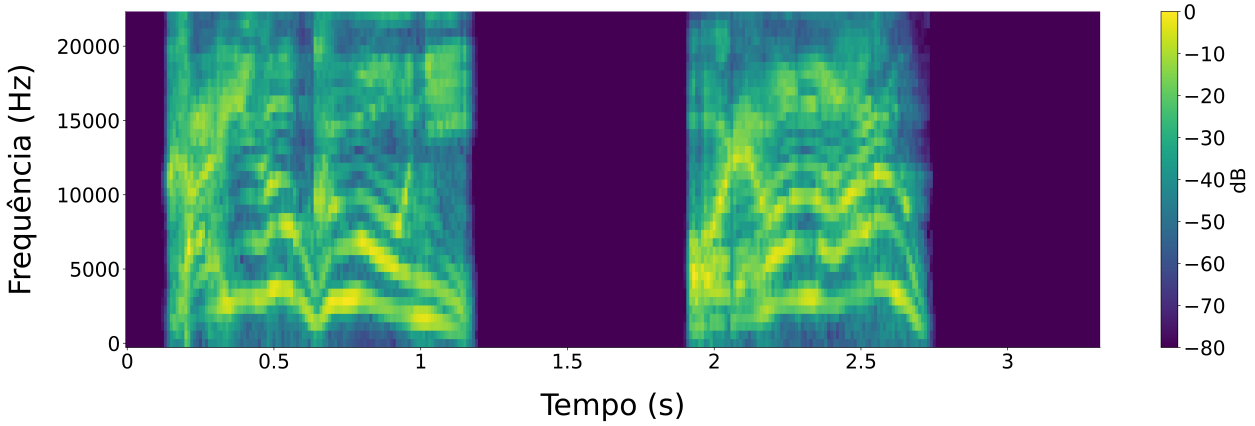


2.3.4 Compressão logarítmica (log-Mel)

As energias das bandas podem diferir por ordens de grandeza, o que faz uma banda dominar a visualização e encobrir detalhes úteis. A compressão logarítmica reequilibra esses valores ao transformar razões multiplicativas em diferenças aditivas. Em potência, os níveis passam a ser expressos em decibéis, sempre relativos a uma referência. Se uma banda tem cem vezes a potência de outra, seu nível fica 20 dB acima. Como a percepção de intensidade cresce de modo aproximadamente logarítmico, essa conversão aproxima a medida da percepção de intensidade [27, 29].

A conversão do mapa de potência Mel para a escala logarítmica em decibéis redistribui os níveis de energia e destaca diferenças relativas entre bandas. Na Figura 13, vê-se o mesmo mapa da Figura 12 após essa conversão, o valor máximo do trecho é tomado como referência e aparece em 0 dB, enquanto valores muito baixos são limitados por um piso de -80 dB. Cores claras indicam níveis próximos de 0 dB e cores escuras aproximam-se desse piso. A compressão reduz a predominância de picos isolados, realça conteúdos de baixa energia e evidencia o contorno relativo entre bandas.

Figura 13 – Log-energia nas bandas Mel em decibéis. Com pico em 0 dB e piso em −80 dB; a compressão realça o contorno relativo entre bandas.



2.3.5 DCT-II e coeficientes cepstrais

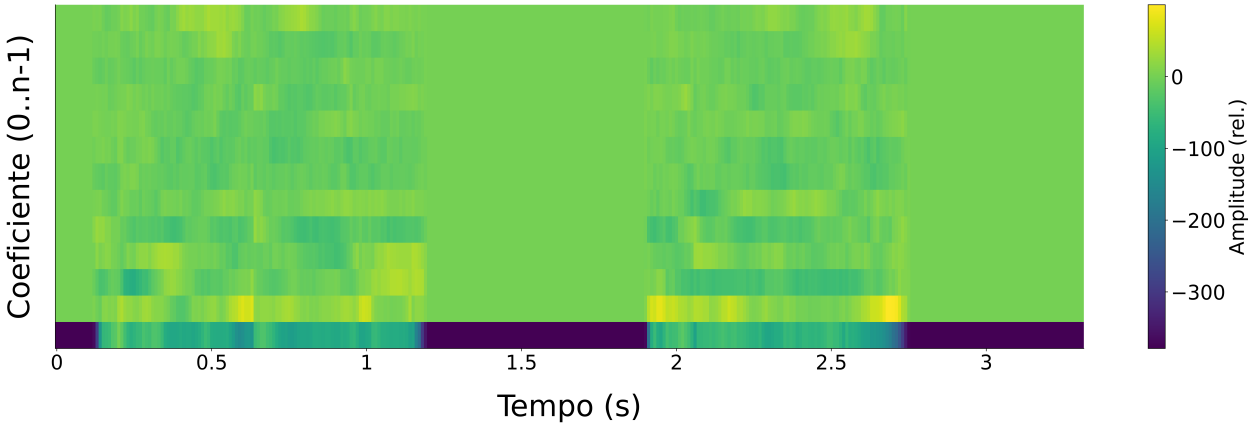
Por fim, aplica-se a DCT-II, *Discrete Cosine Transform*, tipo II, com normalização ortonormal ao vetor de log-energias Mel de cada *frame* [21, 27]. A DCT compara esse vetor a uma família de padrões de cosseno ao longo do eixo de bandas Mel. Esses padrões têm frequência espacial crescente: começam quase planos e passam a oscilar cada vez mais à medida que a ordem p aumenta. Projetar o vetor nesses padrões gera os coeficientes c_p, que medem o grau de semelhança do contorno com cada padrão. A normalização ortonormal preserva a energia, coloca os coeficientes na mesma escala e permite reconstrução exata pela DCT inversa. Como as envoltórias espectrais em música tendem a ser suaves, as ordens baixas concentram a estrutura principal, enquanto as ordens altas registram ondulações finas e voláteis, em geral tratadas como ruído [1, 27].

Nas ordens baixas, os coeficientes descrevem traços amplos da envoltória espectral. O c_0 sintetiza o nível global e corresponde à média das log-energias no *frame*. O c_1 indica a inclinação do espectro ao longo das bandas Mel, refletindo o balanço entre baixas e altas frequências. O c_2 expressa a curvatura, mostrando se há realce das frequências médias em relação às extremas ou o contrário. Na prática, retêm-se apenas os primeiros C coeficientes, por exemplo $C = 13$, e descartam-se ordens mais altas, por serem mais sensíveis a ondulações finas e a flutuações instáveis [21, 27, 1].

Os MFCCs podem ser organizados como um mapa tempo×coeficiente. Cada coluna corresponde a um *frame* do sinal e reúne o vetor de $C = 13$ coeficientes desse instante. Cada linha fixa o índice do coeficiente, com c_0 na base e as ordens seguintes acima. Na Figura 14 esse arranjo aparece como um mapa de cores em que as tonalidades indicam a amplitude do coeficiente em valores relativos, sem unidade de medida, pois derivam da DCT de log-energias sob normalização ortonormal. O aspecto mais relevante é o contorno temporal, em que c_0 despenca nos trechos silenciosos e reflete o nível global. Já c_1 e c_2 variam mais lentamente e acompanham mudanças de inclinação e de curvatura da envoltória. Ademais, as ordens mais altas registram detalhes finos e flutuações rápidas entre *frames*. Em termos dimensionais a

figura representa uma matriz $C \times T$, com $C = 13$ e T igual ao número de *frames*.

Figura 14 – Mapa de calor dos MFCCs por *frame*. Eixo x: tempo, um *frame* por coluna. Eixo y: índice do coeficiente $0, \ldots, C-1$, com c_0 na linha inferior. As cores indicam a amplitude dos coeficientes, sem unidade de medida.



Em síntese, a sequência *forma de onda* → *frames* → *janelamento* → *STFT* → *banco Mel* → *log* → *DCT* transforma uma informação como o da Figura 5 no mapa tempo×coeficientes da Figura 14. Cada etapa preserva o que importa em seu domínio, como a variação temporal, a estabilidade espectral, a organização perceptual, o equilíbrio da escala e a compactação, e resulta numa representação concisa e comparável do timbre [21, 27, 1].

2.4 Classificação de Áudio

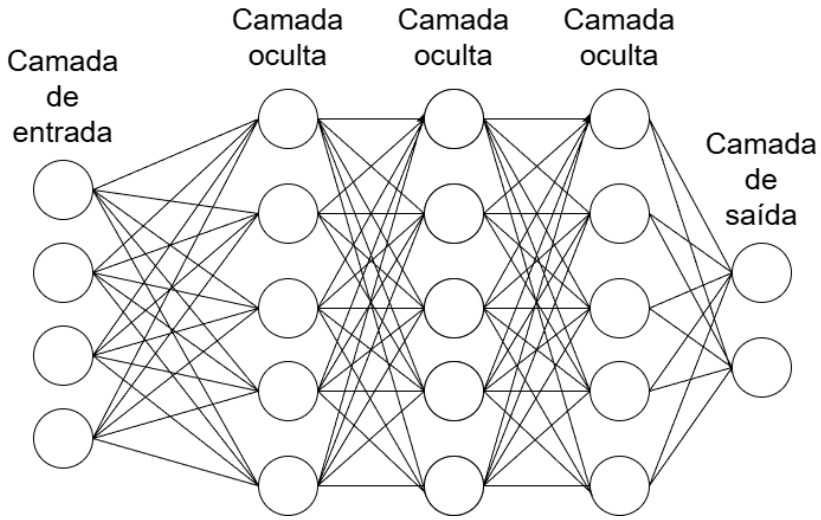
A classificação de áudio consiste na tarefa de atribuir rótulos a trechos de som com base em suas características previamente extraídas. Esses rótulos podem indicar, por exemplo, o gênero musical de uma faixa, os instrumentos que estão presentes na gravação ou até mesmo emoções associadas ao áudio. Um exemplo prático dessa aplicação pode ser observado em sistemas de recomendação de músicas, que utilizam informações sobre o estilo das canções para sugerir playlists personalizadas aos usuários [26, 7].

Entre os classificadores tradicionais que operam sobre vetores de características, destacam-se as redes neurais totalmente conectadas, também conhecidas como perceptrons multicamadas (MLP — *Multilayer Perceptron*) [32, 10]. Essas redes organizam neurônios artificiais em camadas sequenciais, conectando entradas a saídas por meio de múltiplas transformações não lineares.

Essa arquitetura pode ser ilustrada por um diagrama em camadas, no qual os neurônios são organizados em camada de entrada, ocultas e de saída. Uma forma esquemática desse tipo de organização pode ser visualizado na Figura 15. À esquerda observa-se a camada de entrada, que recebe os valores das características extraídas do áudio. No centro ficam as camadas ocultas, responsáveis por realizar transformações não lineares e aprender padrões complexos a partir dos dados. À direita a camada de saída fornece as previsões do modelo, indicando, por exemplo, a probabilidade de cada classe estar presente no áudio. As conexões

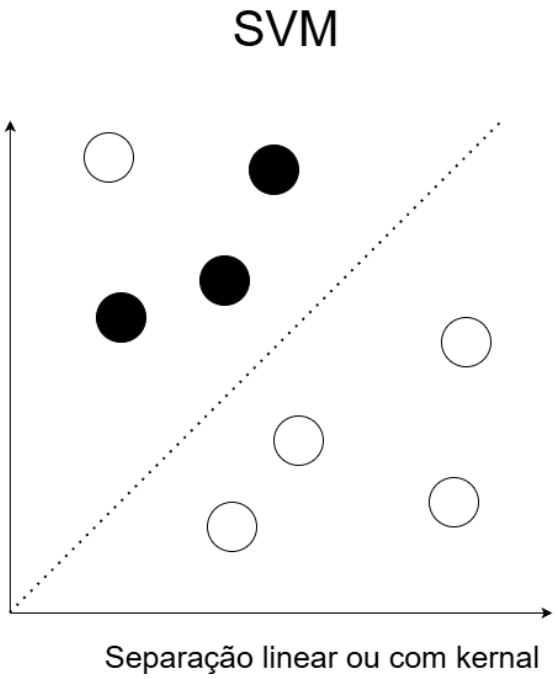
entre neurônios de diferentes camadas são representadas por linhas e demonstram que cada neurônio está conectado a todos os da camada seguinte. Na prática o número de camadas ocultas e de neurônios em cada uma é escolhido conforme a aplicação e as restrições do problema, sem um limite rígido.

Figura 15 – Arquitetura de uma rede neural *fully connected* (perceptron multicamadas): camada de entrada, camadas ocultas e camada de saída.



Dentre as abordagens clássicas está também a Máquina de Vetores de Suporte (SVM — *Support Vector Machine*) [33]. O SVM tenta separar as categorias por uma linha/plano/hiperplano maximizando a *margem*. Quando os dados não são linearmente separáveis, um *kernel* mapeia os exemplos para um espaço em que a separação seja viável. Na Figura 16, a linha pontilhada é o hiperplano ótimo, posicionado para maximizar a distância entre as duas classes, pontos pretos e pontos brancos, e as linhas paralelas indicam as margens.

Figura 16 – Máquinas de Vetores de Suporte (SVM): classificação baseada em separação linear ou com *kernel* entre diferentes categorias.



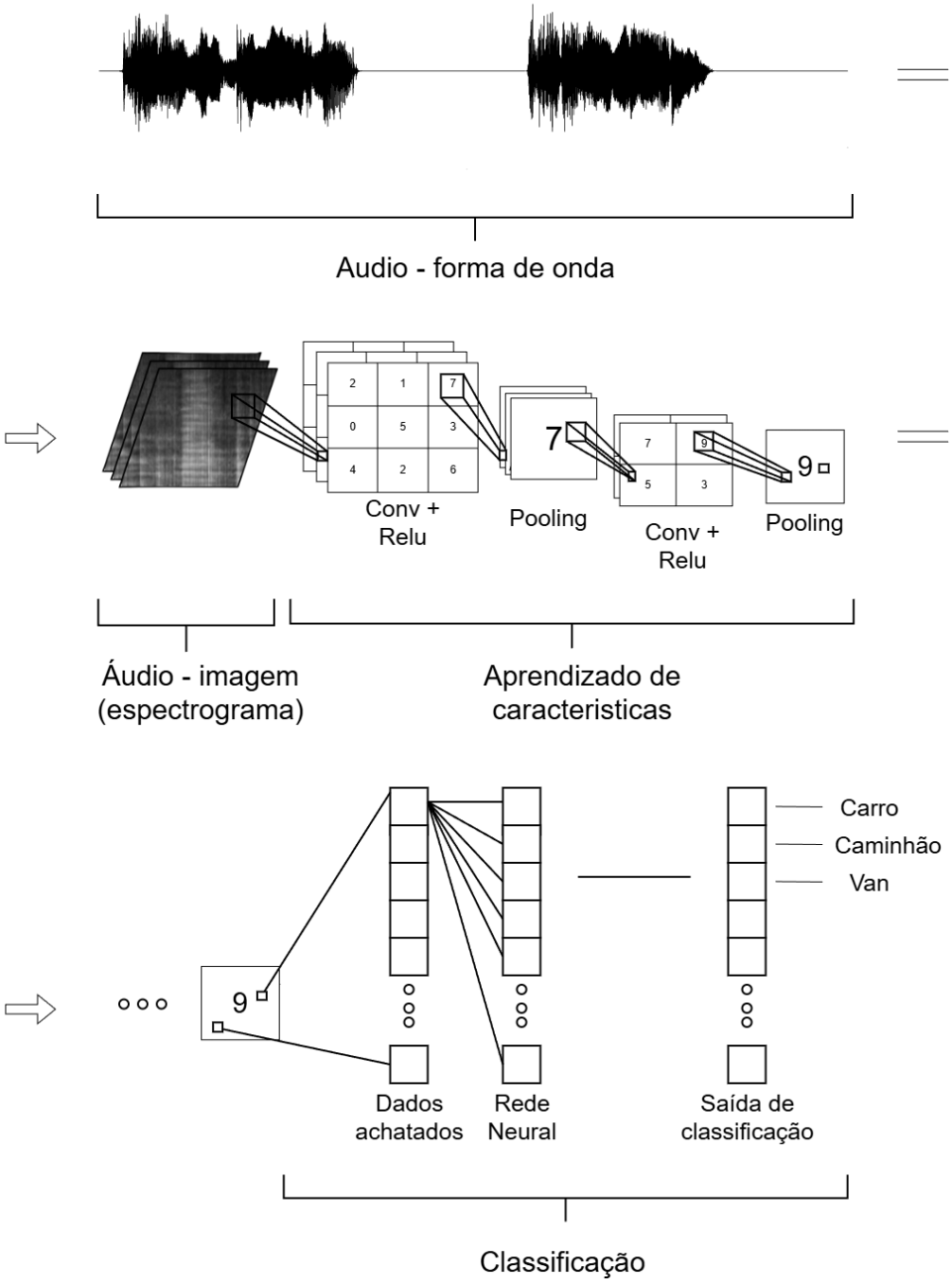
No entanto, à medida que a quantidade de dados cresceu e as tarefas se tornaram mais complexas, abordagens mais sofisticadas, baseadas em redes neurais profundas, passaram a oferecer melhores resultados [34, 7]. Nesse contexto, destacam-se as Redes Neurais Convolucionais (CNNs — *Convolutional Neural Networks*), inicialmente desenvolvidas para o reconhecimento de imagens, mas que também se mostraram muito eficientes no processamento de áudio [34].

Essas redes funcionam como detectores automáticos de padrões, capazes de identificar, por exemplo, sequências harmônicas, batidas rítmicas ou outros aspectos característicos do som. Para isso, as CNNs analisam representações visuais do áudio, como os espectrogramas, que transformam o som em imagens que exibem como as frequências variam ao longo do tempo [7]. Nessa representação, tons de cinza mais claros indicam maior magnitude espectral; em versões coloridas, cores mais intensas desempenham o mesmo papel. Os filtros convolucionais aprendem padrões nesses mapas, como trilhas harmônicas e ataques de percussão.

Um *pipeline* típico de classificação por CNN organiza o processamento do sinal de áudio em estágios sucessivos. O áudio chega como forma de onda e em seguida é convertido em um espectrograma de magnitude, que revela quanta energia há em cada faixa de frequência ao longo do tempo. A partir desse espectrograma inicia-se o processamento por CNN, em que pequenos filtros percorrem a imagem tempo–frequência e respondem quando encontram padrões locais recorrentes. As respostas passam pela função ReLU, que mantém valores positivos e zera os negativos, realçando apenas sinais fortes. Depois aplica-se *pooling*, que resume regiões vizinhas em um único valor, agrega contexto local e diminui a dependência de alinhamento exato no espectrograma. Por fim, os mapas resultantes são achatados e alimentam

camadas densas, que integram as informações extraídas e estimam as probabilidades de cada classe. Esse fluxo pode ser melhor visualizado de forma esquemática na Figura 17.

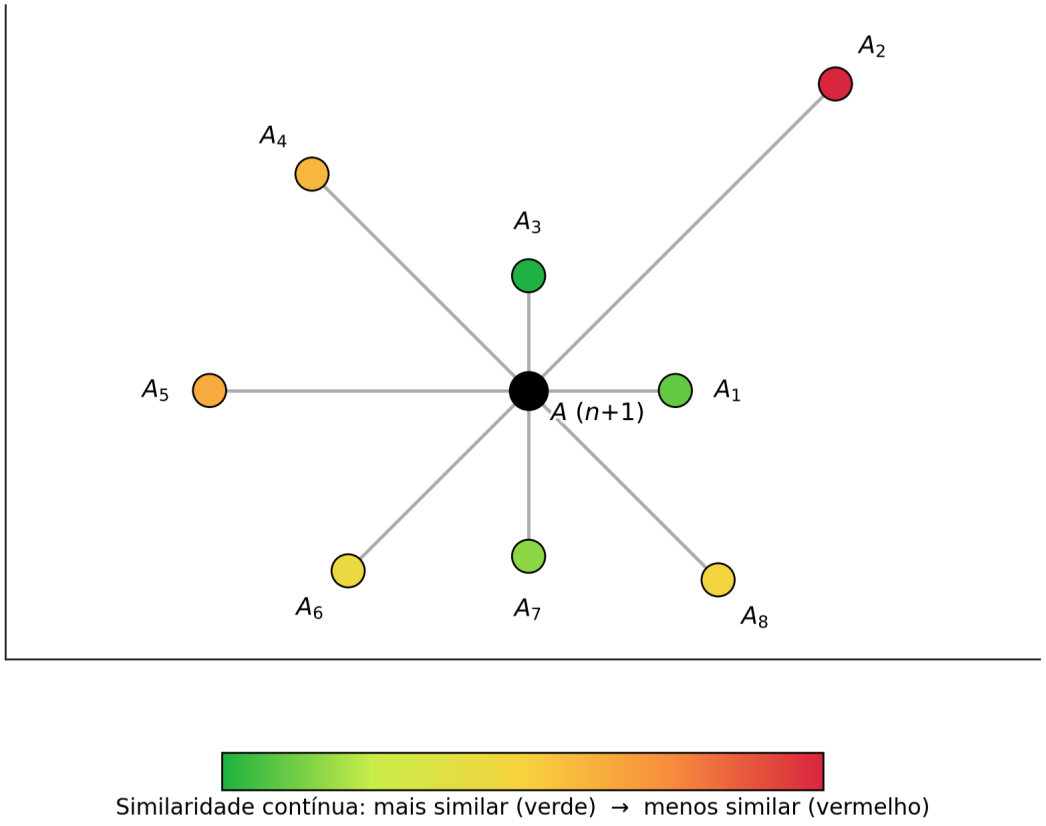
Figura 17 – CNN aplicada a espectrogramas: forma de onda na entrada, conversão para espectrograma de magnitude, convoluções com ReLU que extraem padrões locais, *pooling* que agrega contexto e reduz a dependência de alinhamento exato no espectrograma, seguido de achatamento e camadas densas que estimam as probabilidades de cada classe.



Mais recentemente, novas técnicas de aprendizado, conhecidas como modelos contrastivos, têm sido exploradas para o processamento de áudio. Entre esses modelos, destaca-se o CLMR (*Contrastive Learning of Musical Representations*) [12]. Diferente das abordagens

tradicionais, que dependem de grandes quantidades de exemplos rotulados, o CLMR utiliza uma estratégia de aprendizado auto-supervisionado [12]. A Figura 18 ilustra a etapa de avaliação: após o treinamento contrastivo, cada áudio é mapeado para um *embedding*; itens mais próximos da âncora, neste exemplo o ponto preto central $A^{(n+1)}$, são considerados mais similares.

Figura 18 – CLMR na etapa de avaliação pós-treino. Cada áudio é mapeado a um *embedding*. A âncora $A^{(n+1)}$ é o ponto preto central; quanto menor a distância até esse ponto, maior a similaridade. A cor indica similaridade: verde mais similar e vermelho menos similar.



Para comparar representações sob condições padronizadas de classificação e reduzir interferências de arquitetura, a literatura adota *probes*, isto é, classificadores simples treinados sobre vetores de representação produzidos por um extrator de características fixado durante o treinamento. Para assegurar comparabilidade entre classificadores, a capacidade e o protocolo de treino deles devem permanecer constantes, com preferência por classificadores lineares e, em algumas variações, por classificadores rasos [35, 36, 37, 14]. Além disso, entre os *probes* utilizados com mais frequência, destacam-se o SVM linear e MLPs rasos [14]. Modelos lineares ou rasos são preferidos pela implementação simples e pelo controle direto da capacidade do classificador, assim, o desempenho observado tende a refletir principalmente a qualidade da representação, e não particularidades do classificador [35, 36, 37, 14].

2.5 Bases de Dados

Para que seja possível avaliar e comparar de forma confiável a eficácia e a eficiência de diferentes métodos, é fundamental dispor de bases de dados padronizadas, unificadas e bem documentadas. Nesta subseção, descrevemos alguns conjuntos clássicos e amplamente citados na literatura de MIR, com o objetivo de contextualizar o ecossistema de dados. Parte dessas bases possui acesso facilitado por ferramentas como o `mirdata` [4], enquanto outras são tradicionalmente obtidas por rotinas próprias de *download* e pré-processamento.

Entre as principais bases utilizadas, destaca-se a **GTZAN** (George Tzanetakis's Genre Dataset), uma coleção clássica para avaliação de algoritmos de classificação de gêneros musicais. Ela contém 1.000 arquivos de áudio, cada um com 30 segundos de duração, distribuídos uniformemente entre dez gêneros distintos: blues, classical, country, disco, hip hop, jazz, metal, pop, reggae e rock; com 100 faixas por gênero e taxa de amostragem de 22.050 Hz. Publicada em 2002, a GTZAN tornou-se referência em estudos sobre classificação automática de gêneros musicais [26][1].

Outra base relevante é a **MagnaTagATune**, voltada para tarefas de autotagging e classificação multirrótulo. Possui 25.863 clipes de áudio, cada um com 29 segundos de duração e taxa de amostragem de 16.000 Hz, além de anotações detalhadas, como gênero, instrumentos, timbre, entre outros. Desde sua publicação em 2009, é um recurso fundamental para experimentos envolvendo rotulação automática de músicas e análise de múltiplos atributos sonoros [38][2].

O **NSynth** (Neural Synthesizer Dataset), desenvolvido pelo projeto Magenta do Google, reúne 305.979 sons sintéticos e reais de instrumentos musicais. Cada amostra tem 4 segundos de duração, com taxa de amostragem de 16.000 Hz, e inclui metadados como pitch, instrumento e envelope. Lançado em 2017, o NSynth é amplamente utilizado em pesquisas relacionadas à síntese de áudio e classificação de timbres instrumentais [39][3].

Por fim, a base **MTG-Jamendo** contém aproximadamente 55.000 músicas completas, sob licença Creative Commons, anotadas de forma multirrótulo com base em metadados sociais como gênero, tema, e emoção, entre outros. Com taxa de amostragem de 44.100 Hz e duração variável das faixas, essa base, publicada em 2019, permite estudos em larga escala sobre classificação multirrótulo e análise de conteúdo musical diversificado [40][4].

Para facilitar a comparação, a Tabela 1 apresenta um resumo das principais características dessas bases. A coluna *Base* identifica cada conjunto de dados; *Ano* indica o ano de publicação; *Qtde Áudios* informa o número total de faixas; *Duração* é o tempo típico por faixa; *Amostragem* indica a taxa de amostragem em Hz; e *Tipo de Anotação* resume o alvo principal, por exemplo: gêneros, instrumentos ou múltiplas tags.

1 Disponível em: http://marsyas.info/downloads/datasets.html. Acessado em 04/11/2025.
2 Disponível em: https://mirg.city.ac.uk/codeapps/the-magnatagatune-dataset. Acessado em 04/11/2025.
3 Disponível em: https://magenta.tensorflow.org/datasets/nsynth. Acessado em 04/11/2025.
4 Disponível em: https://mtg.github.io/mtg-jamendo-dataset/. Acessado em 04/11/2025.

Tabela 1 – Comparação entre as bases de dados utilizadas

Base	**Ano**	**Qtde Áudios**	**Duração**	**Amostragem**	**Tipo de Anotação**
GTZAN	2002	1.000	30 s	22.050 Hz	Gênero (10 classes)
MagnaTagATune	2009	25.863	29 s	16.000 Hz	Multirrótulo (tags diversas)
NSynth	2017	305.979	4 s	16.000 Hz	Instrumento, pitch, envelope
MTG-Jamendo	2019	55.000	variável	44.100 Hz	Multirrótulo (gênero, emoção, tema)

2.6 Ferramentas de Avaliação

A etapa de avaliação é essencial para medir o desempenho dos modelos aplicados em tarefas de MIR. Sem uma avaliação apropriada, torna-se difícil saber se um modelo realmente aprendeu a resolver o problema proposto. E até mesmo se os resultados obtidos ocorreram ao acaso. Por isso, o uso de medidas padronizadas permitem comparar diferentes abordagens de forma justa, garantindo a reprodutibilidade e a autenticidade dos experimentos.

O `mir_ref` disponibiliza um conjunto de medidas bem conhecidas na área de aprendizado de máquina, especialmente em problemas de classificação multirrótulos, como o autotagging. As principais medidas calculadas pela ferramenta são a AUC-ROC (Área sob a Curva ROC, do inglês *Receiver Operating Characteristic*) [41] e a Precisão Média (Average Precision) [42]. A seguir, são apresentadas cada uma delas:

- **AUC-ROC (Área sob a Curva ROC — *Receiver Operating Characteristic*):** mede a capacidade de o modelo separar corretamente os exemplos positivos dos negativos [41]. A curva ROC é construída ao variar o limiar de decisão do classificador, mostrando como mudam a taxa de verdadeiros positivos (*True Positive Rate* — TPR) e a taxa de falsos positivos (*False Positive Rate* — FPR). A AUC (Área Sob a Curva) resume essa informação em um único número entre 0 e 1: quanto mais próximo de 1, melhor o desempenho do classificador; 1 representa um classificador perfeito, 0,5 indica um modelo que decide aleatoriamente, e valores abaixo de 0,5 indicam desempenho pior que o acaso, ou seja, o modelo erra sistematicamente a classificação.

  As taxas TPR e FPR são definidas como:

  $$\text{TPR} = \frac{TP}{TP + FN}, \quad \text{FPR} = \frac{FP}{FP + TN}$$

  onde TP representa os verdadeiros positivos, FP os falsos positivos, FN os falsos negativos e TN os verdadeiros negativos.

  Um **verdadeiro positivo (TP)** ocorre quando o modelo prediz a presença de um rótulo que de fato está presente; **falso positivo (FP)**, quando prediz presença, mas o

rótulo não está presente; **falso negativo (FN)**, quando deixa de predizer um rótulo que está presente; e **verdadeiro negativo (TN)**, quando corretamente prediz a ausência do rótulo.

Exemplo numérico: considere que o modelo, ao longo de diferentes limiares, apresentou uma curva ROC cuja AUC foi calculada em 0,92. Esse valor pode ser interpretado como a probabilidade de um exemplo positivo receber pontuação maior do que um negativo escolhido ao acaso; assim, AUC = 0,92 indica que, em média, há 92% de chance de o modelo ranquear um positivo acima de um negativo aleatório [41].

- **Precisão Média (Average Precision — AP):** é uma medida que resume a curva de precisão-revocação em um único valor, refletindo a qualidade das predições ao longo de diferentes limiares de decisão [43]. A precisão (*precision*) indica, entre todas as vezes que o modelo previu um determinado rótulo como presente, quantas vezes ele acertou. A revocação (*recall*) indica, entre todas as vezes que esse rótulo realmente estava presente, quantas vezes o modelo conseguiu identificá-lo.

  A Precisão Média corresponde à soma ponderada das precisões obtidas em cada nível de revocação, ou seja, calcula-se a área sob a curva de precisão-revocação de forma discreta:

$$\text{Average Precision} = \sum_n (R_n - R_{n-1}) P_n$$

  onde P_n e R_n representam, respectivamente, a precisão e a revocação no ponto n da curva. Note que se trata de uma soma, que aproxima numericamente a área sob a curva, em contraste com a integral definida de funções contínuas.

  Exemplo numérico: suponha que, em diferentes pontos da curva, o modelo apresente os seguintes valores de precisão e revocação apresentados na Tabela 2:

Tabela 2 – Exemplo de pontos da curva precisão–revocação usados no cálculo da Average Precision.

Revocação (Recall)	Precisão (Precision)
0.1	0.90
0.4	0.75
0.7	0.60
1.0	0.50

  Aplicando a soma discreta da área sob a curva (com $R_0 = 0$):

$$\begin{aligned}
\text{AP} &= (0.1 - 0) \cdot 0.90 + (0.4 - 0.1) \cdot 0.75 + (0.7 - 0.4) \cdot 0.60 + (1.0 - 0.7) \cdot 0.50 \\
&= 0.10 \cdot 0.90 + 0.30 \cdot 0.75 + 0.30 \cdot 0.60 + 0.30 \cdot 0.50 \\
&= 0.090 + 0.225 + 0.180 + 0.150 \\
&= \mathbf{0.645}.
\end{aligned}$$

Assim, a *Average Precision* deste exemplo é 0.645 (64,5%).

2.7 Plataformas e *Frameworks*

Diversas plataformas e ferramentas têm sido desenvolvidas ao longo dos anos para apoiar a pesquisa em MIR, atendendo a diferentes necessidades e etapas do fluxo de trabalho científico. Cada uma delas foca em aspectos específicos do processamento musical, como extração de características, avaliação padronizada, *benchmarking* de algoritmos, ou manipulação flexível de sinais de áudio.

Entre as principais iniciativas, destacam-se:

- **MIREX:** iniciativa comunitária responsável por organizar *benchmarks* anuais para algoritmos de MIR em múltiplas tarefas, promovendo comparabilidade e avanço coletivo na área [18].
- **MARBLE:** *benchmark* abrangente que propõe um protocolo unificado para avaliação sistemática e imparcial de representações musicais [14].
- **MIRFLEX:** biblioteca modular voltada à extração de diferentes características musicais, como ritmo, tonalidade e timbre, com forte ênfase em flexibilidade e reutilização de componentes [19].
- **Essentia:** biblioteca em C++/Python amplamente utilizada para análise de áudio, oferecendo um vasto conjunto de algoritmos para extração de características, segmentação e classificação musical, sendo referência em soluções para sistemas de recomendação [1].
- **LibROSA:** biblioteca Python muito popular entre pesquisadores e engenheiros de dados, projetada para facilitar a análise, manipulação e transformação de sinais de áudio, e facilmente integrável a *frameworks* de aprendizado de máquina [27].

A disponibilidade dessas plataformas tem impulsionado a evolução do campo, permitindo que pesquisadores comparem abordagens sob condições controladas e repliquem experimentos de forma mais transparente. No entanto, apesar do grande avanço proporcionado por essas ferramentas, muitas vezes elas atuam de maneira fragmentada, com integrações parciais entre etapas de um *pipeline* ou exigindo customização significativa por parte do usuário para garantir a reprodutibilidade dos experimentos.

Na prática, essas iniciativas tendem a cobrir apenas partes desse fluxo. Ferramentas como a Essentia e a LibROSA concentram-se na extração e manipulação de características, deixando a cargo do pesquisador a definição dos particionamentos, o treinamento dos classificadores e a comparação sistemática entre representações [1, 27]. O MIREX organiza campanhas anuais de avaliação de algoritmos de MIR em tarefas específicas, como classificação de gênero, similaridade musical e extração de melodia, entre outras, e fornece conjuntos de dados e protocolos de avaliação padronizados, mas não disponibiliza um conjunto de ferramentas

integrado que o pesquisador possa instalar e usar para conduzir esses experimentos localmente [18]. O MARBLE propõe um protocolo abrangente de *benchmarking* de representações, porém pressupõe que o usuário integre, por conta própria, bibliotecas de extração, bases de dados e rotinas de treinamento para colocar o protocolo em prática [14].

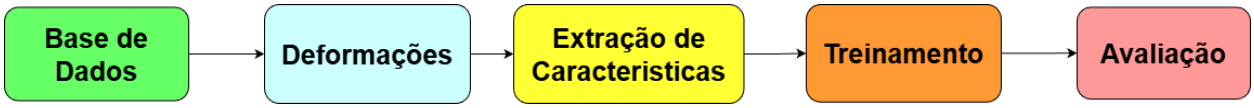
2.8 `mir_ref`

Apesar das contribuições relevantes dessas iniciativas, muitas ainda não oferecem uma estrutura plenamente integrada que permita conduzir experimentos combinando múltiplas tarefas, representações, distorções de áudio e configurações de experimentos em um mesmo ambiente padronizado. Esse cenário evidencia a necessidade de ferramentas unificadas e extensíveis, capazes de abranger todas essas dimensões de forma centralizada, motivando o surgimento de soluções como a biblioteca `mir_ref`, abordada neste trabalho [20].

O `mir_ref`, proposto por Plachouras *et al*, é um *framework* voltado para a centralização, padronização e simplificação da avaliação de representações de áudio em tarefas de MIR. A plataforma reúne, em um mesmo ambiente, a organização dos dados, a extração de diferentes tipos de representações, o treinamento de classificadores com protocolos padronizados, a análise de robustez, o cálculo de medidas de avaliação e a consolidação dos resultados em relatórios. Essa integração coesa e flexível facilita a comparação entre métodos clássicos e modernos. Ademais, o *framework* foi concebido para que novos extratores de características possam ser acoplados ao *pipeline* e avaliados sob as mesmas condições que representações já presentes.

Além disso, a operação do `mir_ref` se dá por meio de um *pipeline* estruturado em cinco etapas principais, conforme ilustrado na Figura 19. A primeira etapa consiste na seleção de um conjunto de dados, a partir do qual os áudios são organizados e preparados. Opcionalmente, na segunda etapa, pode-se aplicar deformações nos sinais de áudio, como ruído, compressão ou alteração de *pitch*[5], com o objetivo de simular condições adversas e testar a robustez dos métodos de representação frente a variações comuns em contextos reais de uso. Em seguida, na terceira etapa, realiza-se a extração de características como *embeddings* ou descritivos espectrais que alimentam a etapa de treinamento, composta por classificadores padronizados. Por fim, os modelos são avaliados com base em medidas específicas para a tarefa proposta.

Figura 19 – Fluxo de execução do *framework* `mir_ref`, destacando suas principais etapas: seleção do conjunto de dados, deformação (opcional), extração de características, treinamento e avaliação.



Em termos de modelos de representações de áudio, o `mir_ref` já disponibiliza um conjunto diverso de *embeddings* neurais pré-treinados: CLMR, VGGish, OpenL3 e MusicNN.

5 *Pitch* é a altura percebida de um som, relacionada à frequência fundamental do sinal.

Há ainda modelos adicionais treinados com metadados do Discogs, uma base colaborativa de lançamentos fonográficos, artistas e estilos, como EffNet/Discogs e MAEST [44, 45, 46]; o NeuralFP, impressão digital neural para identificação robusta de trechos curtos [47, 48]; e o MERT, modelo auto-supervisionado de grande porte, que permite extrair vetores de diferentes camadas internas [49].

3 Proposta e Metodologia

Este trabalho propõe integrar o MFCC ao `mir_ref`, disponibilizando-o como método de referência reprodutível para qualquer tarefa suportada pelo *framework*, permitindo, por exemplo, comparar diretamente *embeddings* neurais com os mesmos particionamentos, *probes* e medidas de avaliação, reduzindo a quantidade de código específico que o pesquisador precisa escrever e diminuindo o risco de variações experimentais difíceis de reproduzir [20]. A extensão implementada adiciona um novo caminho de extração de características, configurável por meio do arquivo de configuração YAML, e compatível com o restante do fluxo operacional. Ademais, como forma de validar e exemplificar essa integração em uso, a Seção 4 apresenta um estudo de caso em que os MFCCs são comparados com representações provenientes de redes neurais profundas, como CLMR e VGGish.

Esta seção descreve a metodologia adotada para essa integração: ferramentas empregadas, parametrização escolhida e funcionamento da implementação que torna a extração de MFCCs compatível com o fluxo operacional do `mir_ref`. A versão do `mir_ref` estendida com suporte aos MFCCs, desenvolvida neste trabalho, está disponível no link https://github.com/JoaoSartoreto/mir_ref. Um guia passo a passo de sua execução encontra-se no Apêndice A.

3.1 Descrição da Proposta

Os MFCCs foram escolhidos como foco deste estudo por constituírem uma das representações clássicas mais consolidadas na literatura de processamento de áudio e MIR. Além disso, a ausência dessa técnica no conjunto de representações originalmente implementadas no `mir_ref` motivou sua inclusão, de modo a suprir essa lacuna e permitir uma comparação sistemática com abordagens modernas de representação.

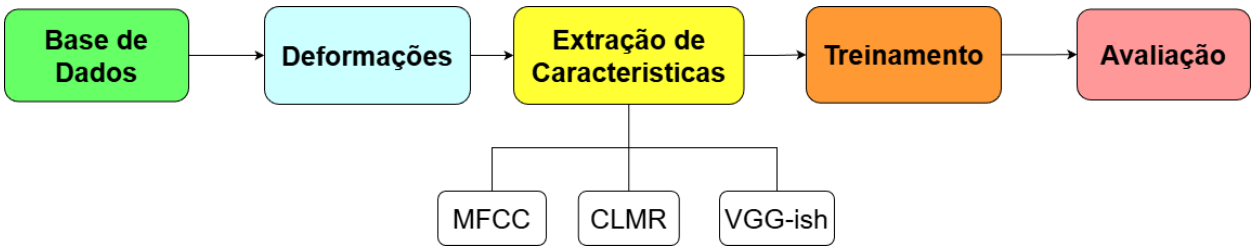
Essa popularidade advém da capacidade dos MFCCs de representar a forma do espectro de potência de um sinal de áudio utilizando uma escala de frequência baseada na percepção auditiva humana, conhecida como frequência Mel. Dizer que essa escala é "perceptivamente motivada" significa que ela foi desenvolvida para refletir como o ouvido humano percebe variações de frequência, ou seja, enfatizando faixas de frequência para as quais somos mais sensíveis e comprimindo regiões onde nossa percepção é menos detalhada. Assim, os MFCCs conseguem capturar informações relevantes sobre o conteúdo espectral do áudio com baixo

custo computacional, sendo amplamente adotados em tarefas como reconhecimento de fala, classificação de gêneros musicais, detecção de instrumentos, entre outras aplicações [21].

A integração do MFCC respeitou a estrutura do `mir_ref`, que exige o armazenamento das características em vetor unidimensional obtido por *flattening*, isto é, a conversão da matriz de coeficientes ao longo do tempo em um único vetor, o que permite seu uso nas etapas subsequentes de treinamento e avaliação. Para verificar a consistência dessa implementação, foram conduzidos experimentos que avaliaram a eficácia dos MFCCs em relação aos *embeddings* gerados pelos modelos CLMR e VGGish, com o objetivo de possibilitar uma comparação sistemática e reprodutível entre as representações.

A etapa de extração de características pode ser realizada por meio de diferentes métodos como MFCC, CLMR, VGGish, ou qualquer outro modelo compatível antes do treinamento e avaliação dos mesmos. Essa organização padroniza o processo e facilita a comparação direta entre diferentes representações de áudio. Para melhor entendimento, a Figura 20 retoma o fluxo de execução do `mir_ref`, apresentado na subseção dedicada ao `mir_ref` em Conceitos Relacionados, mas, agora, enfatizando o caráter configurável dessa etapa.

Figura 20 – Fluxo de execução do `mir_ref`, evidenciando a integração dos MFCCs como uma das opções disponíveis na etapa de extração de características, juntamente com modelos como CLMR e VGGish. Outros métodos também podem ser incorporados ao *pipeline* do `mir_ref`.



3.2 Ferramentas Utilizadas

A implementação da extração de MFCCs foi desenvolvida utilizando a linguagem Python, em razão de sua ampla adoção na comunidade de ciência de dados e aprendizado de máquina, bem como pela disponibilidade de bibliotecas especializadas em processamento de sinais. Para a etapa de extração, utilizou-se a biblioteca `librosa` [27], ferramenta reconhecida por sua robustez e facilidade na manipulação de sinais de áudio, além de oferecer suporte nativo para a extração dos MFCCs e demais características espectrais [27, 1]. Cabe notar que o `mir_ref` é implementado em Python, o que reforça essa escolha e simplifica a integração.

A escolha da `librosa` se justifica pela sua interface de alto nível, que facilita a leitura de diferentes formatos de arquivos de áudio, o processamento direto em arrays NumPy e a adaptação automática à taxa de amostragem (*sample rate*) presente em cada arquivo [27]. Esse aspecto é útil neste trabalho, que utiliza bases de dados musicais acessadas por meio

da biblioteca `mirdata`, bastante empregada na área para o gerenciamento, *download* e padronização do acesso a conjuntos de dados de áudio [4]. Essas bases podem apresentar arquivos com diferentes especificações técnicas, e a compatibilidade da `librosa` com múltiplos formatos e taxas de amostragem contribui para o processamento correto de todos os arquivos.

Adicionalmente, o desenvolvimento foi realizado respeitando a estrutura modular do `mir_ref`, garantindo que a implementação da extração dos MFCCs possa ser incorporada ao fluxo de execução de experimentos já existente, possibilitando a execução automática de diferentes testes sem a necessidade de alterações manuais no código principal do *framework* por cada pesquisador que quiser usar o MFCC em seus experimentos.

3.3 Configuração dos Experimentos

O objetivo desta etapa foi investigar o impacto de diferentes configurações dos parâmetros dos MFCCs na tarefa de classificação musical, utilizando o `mir_ref`. Para a análise comparativa entre os diferentes cenários, o processo experimental foi automatizado, com logs e arquivos de saída sendo organizados conforme as combinações de parâmetros avaliadas.

3.3.1 Seleção dos Parâmetros do MFCC

A escolha dos parâmetros dos MFCCs seguiu padrões adotados pela maioria dos principais trabalhos encontrados na literatura de MIR e foi complementada por uma análise empírica neste estudo. Os parâmetros definidos com base na literatura, foram:

Taxa de amostragem (`sample_rate`): manteve-se a taxa original de cada arquivo, conforme sugerido por McFee et al. [27] e reforçado por Tzanetakis e Cook [26], evitando distorções ou perdas devidas a reamostragem, sobretudo em conjuntos heterogêneos.

Número de coeficientes MFCC (`n_mfcc`): quantidade de coeficientes cepstrais devolvidos pela DCT; valores 13 e 16 são amplamente adotados por permitirem uma representação compacta das características espectrais relevantes [50, 51, 52, 53, 21, 54].

Tamanho da janela (`window_size_ms`): duração temporal analisada por *frame*; 20 ms é um compromisso clássico entre resolução temporal e frequência [52, 51, 55, 21].

Avanço entre janelas (`hop_length`): deslocamento entre janelas consecutivas; adotou-se `hop_length =n_fft//2`. O operador `//` indica divisão inteira — calcula $n_{\text{fft}}/2$ e descarta a parte fracionária. Assim, obtém-se uma sobreposição de 50% entre *frames*, prática consolidada [54, 52, 56, 57, 21].

Número de filtros Mel (`n_mels`): tamanho do banco de filtros na frequência Mel; 26 e 40 são recorrentes para fala e tarefas musicais, respectivamente [54, 56, 58, 21].

Frequência mínima (`fmin`): limite inferior da faixa analisada pelos filtros Mel; foram considerados 20 Hz, limite inferior da audição humana [54, 21], e 0 Hz [59].

Frequência máxima (`fmax`): limite superior da faixa analisada; 8000 Hz cobre a maior parte do conteúdo relevante para música e voz humana [54, 60, 61, 21].

Tipo de DCT (`dct_type`): projeção no domínio cepstral; adotou-se DCT-II, padrão em implementações modernas [54, 27, 62].

Normalização (`norm`): esquema de normalização da DCT; utilizou-se *ortho*, favorecendo estabilidade numérica e comparabilidade [54, 27, 62, 21].

As faixas testadas por parâmetro estão resumidas na Tabela 3. Por viabilidade computacional, limitou-se a abrangência de valores avaliados.

Tabela 3 – Faixas dos parâmetros dos MFCCs testados nos experimentos.

Parâmetro	**Valores testados**	**Referências**
n_mfcc	13, 16	[50, 51, 52, 53, 54]
window_size_ms	20	[52, 51, 55]
hop_length	n_fft // 2	[54, 52, 56, 57]
n_mels	26, 40	[54, 56, 58]
fmin	20, 0	[54, 59]
fmax	8000	[54, 60, 61]
dct_type	DCT-II	[54, 27]
norm	ortho	[54, 27]

Para selecionar os melhores parâmetros, foi realizada uma fase inicial de experimentação, testando diferentes configurações no contexto do problema. Entretanto, além dessa experimentação, buscou-se embasamento em parâmetros conhecidos da literatura, considerados eficazes em tarefas similares, para reduzir a quantidade de testes com potencial de resultados subótimos [21, 27]. Por meio deste processo, definiu-se o conjunto final de valores que proporcionou melhor desempenho nos experimentos. Os parâmetros, definidos diretamente no arquivo de configuração de exemplo, estão apresentados na Tabela 4.

Tabela 4 – Parâmetros finais utilizados para extração dos MFCCs.

Parâmetro	**Valor final**	**Descrição**
`n_mfcc`	13	Número de coeficientes cepstrais
`window_size_ms`	20 ms	Tamanho da janela
`hop_length`	`n_fft // 2`	Deslocamento entre janelas
`n_mels`	40	Filtros na frequência Mel
`fmin`	20 Hz	Frequência mínima
`fmax`	8000 Hz	Frequência máxima
`dct_type`	2	Tipo da DCT (DCT-II)
`norm`	*ortho*	Normalização ortonormal
`max_frames`	250	Número máximo de *frames*

A automação dos experimentos foi implementada exclusivamente para este estudo e não integra a versão do *framework* `mir_ref` alterada neste trabalho. No código disponibili-

zado, a extração dos MFCCs utiliza a configuração com melhor desempenho observada nos experimentos; ainda assim, o usuário pode definir outros valores de parâmetros por meio do arquivo de configuração `example.yml`.

Diferente de estudos que fixam a taxa de amostragem (*sample rate*), neste trabalho optou-se por manter o *sample rate* original dos áudios presentes em cada base de dados. Essa decisão visou preservar as características originais dos sinais e evitar distorções decorrentes do reamostramento, conforme apontado por McFee et al. [27].

3.3.2 Padronização temporal e pipeline de extração

Como o número de *frames* pode variar conforme a duração do áudio, adotou-se a padronização temporal para `max_frames`, por amostragem ou preenchimento.

As etapas do *pipeline* de extração dos MFCCs são as seguintes:

1. **Carregamento do áudio:** O arquivo é lido na taxa de amostragem original.
2. **Normalização:** O sinal é normalizado em amplitude.
3. **Segmentação em *frames*:** O áudio é dividido em janelas de 20 ms com sobreposição de 50%.
4. **Extração dos MFCCs:** Calcula-se `mfcc` com os parâmetros definidos na Tabela 3.
5. **Normalização dos coeficientes:** Os valores dos MFCCs são escalados para o intervalo [0, 1].
6. **Ajuste do número de *frames*:** O total de *frames* por áudio é ajustado para 250.
7. ***Flattening*:** O resultado é convertido em vetor unidimensional.
8. **Armazenamento:** O vetor é salvo para uso posterior no *pipeline*.

Além dos parâmetros acústicos, avaliou-se também o impacto do número de *frames* utilizado após a extração dos MFCCs, etapa que chamamos de padronização temporal. A extração percorre toda a duração do áudio e, por isso, geralmente produz mais *frames* do que o valor final definido em `max_frames`. Por exemplo, com a configuração `n_mfcc`=13, `n_mels`=40, `fmin`=20 Hz, `fmax`=8000 Hz, `dct_type`=2, `norm`=*ortho*, `window_size_ms`=20 ms e `hop_length`=`n_fft//2`, um arquivo típico resultou em 2913 *frames*. Como a etapa de treinamento do `mir_ref` opera sobre vetores unidimensionais, esse volume implicaria um custo inviável de processamento e memória. Para lidar com isso, fixou-se `max_frames` por áudio: se a extração gerar mais *frames* que o limite, selecionam-se `max_frames` *frames* igualmente espaçados ao longo da sequência; se gerar menos, completa-se com zeros até atingir `max_frames`, procedimento conhecido como *zero-padding*. Neste trabalho, chamam-se *parâmetros* as escolhas internas do cálculo dos MFCCs, como `n_mfcc`, `n_mels`, `fmin`, `fmax`, tipo de DCT e normalização; e *hiperparâmetro* a decisão externa que regula o uso do vetor ao longo

do *pipeline*, no caso `max_frames`. Para estudar o efeito desse hiperparâmetro, consideraram-se os valores 5, 7, 10, 15, 20, 30, 50, 100, 250, 500. Para cada valor de `max_frames`, executaram-se 10 rodadas; cada rodada é composta por 8 execuções, uma para cada configuração de parâmetros de MFCC. No total, isso perfaz 800 execuções (10 `max_frames` × 10 rodadas × 8 configurações).

3.3.3 Configuração do treinamento

Durante a implementação do *pipeline* experimental, foi necessário garantir que os dados extraídos estivessem no formato apropriado compatível com o *framework* `mir_ref`, em particular por meio da conversão dos MFCCs utilizando a operação de *flattening*, que resulta em um vetor unidimensional. Inicialmente, observou-se uma sobrecarga inesperada de recursos computacionais durante o treinamento, que foi posteriormente solucionada ao ajustar corretamente o processo de *flattening* dos dados. Com isso, tornou-se possível realizar os experimentos, priorizando a análise das configurações mais comuns na literatura.

Na etapa de treinamento não foi fixada uma semente, parâmetro que controla a aleatoriedade e, quando fixado, torna reprodutíveis escolhas como a inicialização dos pesos e a ordem de apresentação dos exemplos, de modo que cada execução inicia com pesos distintos e com nova ordem de apresentação dos exemplos a cada época, o que gera flutuações naturais nas medidas de avaliação. Os particionamentos entre conjuntos de treino, validação e testes permaneceram constantes em todas as execuções, assim, as diferenças sistemáticas entre configurações refletem sobretudo as escolhas de extração, enquanto as variações entre repetições de uma mesma configuração correspondem à aleatoriedade do processo de treino. As medidas são reportadas como médias e desvios-padrão ao longo dessas repetições.

4 Resultados

Esta seção apresenta os resultados obtidos nos experimentos realizados conforme a metodologia descrita anteriormente. São exibidas as medidas de desempenho para cada configuração testada dos MFCC, bem como a comparação com as representações modernas CLMR e VGGish. Em todos os casos, consideramos o cenário limpo, entendido como uso dos áudios em sua forma original, sem adição de ruído e sem modificações no sinal. Os arquivos do MagnaTagATune foram utilizados tal como disponibilizados, e as partições de treino, validação e teste permaneceram fixas.

Nas análises entre configurações de MFCC, são reportadas as média das medidas AUC-ROC e AP de cada configuração específica. Essa média foi calculada a partir de 10 execuções independentes por configuração. O desenho experimental foi o seguinte: para o hiperparâmetro `max_frames` foram avaliados os valores 5, 7, 10, 15, 20, 30, 50, 100, 250 e 500; para cada valor foram realizadas 10 rodadas; em cada rodada foram treinadas 8 configurações de parâmetros de MFCC. Assim, cada configuração conta com 10 execuções,

e o valor reportado para ela corresponde à média dos seus 10 resultados. Na comparação entre representações, que envolve MFCC na melhor configuração, CLMR e VGGish, foram reportados a média e o desvio-padrão das medidas AUC-ROC e AP obtidos em 10 execuções independentes para cada representação.

4.1 Resultados dos MFCC

Para avaliar o impacto dos parâmetros na qualidade das representações extraídas, foram variados, o número de coeficientes `n_mfcc`, a quantidade de filtros na frequência Mel `n_mels`, o limite de frequência inferior `fmin` e o número de janelas `max_frames`. As medidas AUC-ROC e AP foram reportadas apenas como médias por configuração. Os particionamentos de treino, validação e teste permaneceram fixos em todas as execuções. A variação entre execuções ocorreu exclusivamente devido à semente aleatória, que define a inicialização dos pesos e a ordem de apresentação dos exemplos a cada época.

Em cada configuração, foram treinados três classificadores rasos padronizados usados para avaliar a qualidade da representação, com níveis de complexidade distintos: Modelo 0, perceptron simples sem camada oculta; Modelo 1, perceptron com uma camada oculta cujo número de neurônios é igual à dimensão do vetor de entrada; Modelo 2, perceptron com duas camadas ocultas com 256 e 128 neurônios. Cada um desses classificadores é identificado nas tabelas e gráficos como “Modelo 0”, “Modelo 1” e “Modelo 2”, respectivamente.

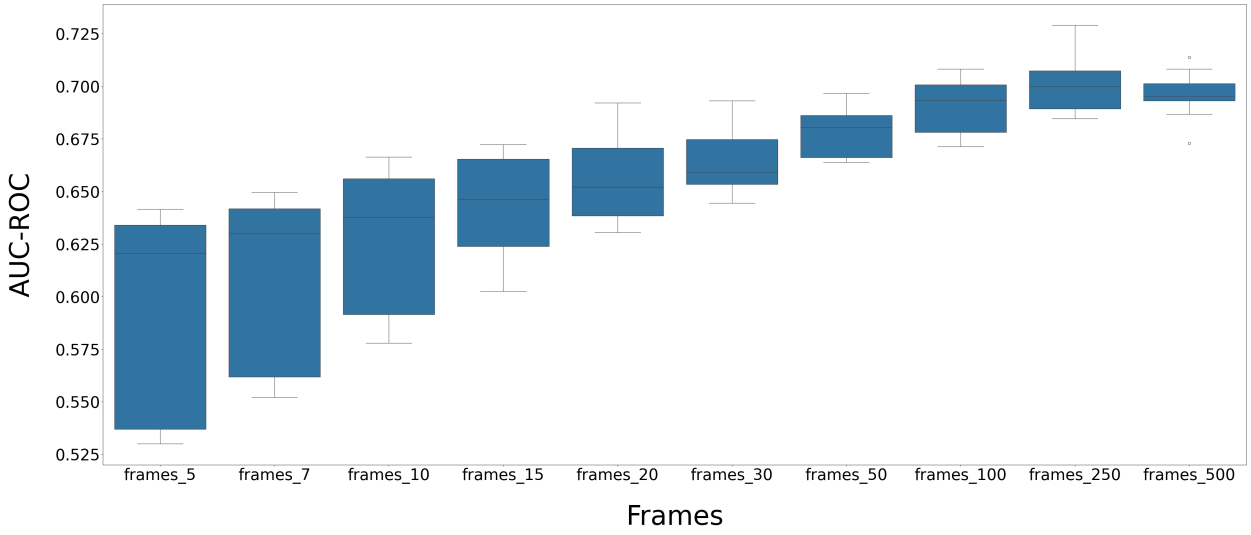
A Tabela 5 apresenta as 10 configurações com melhor desempenho médio no cenário limpo, sem modificações no sinal, destacando a combinação “13 coeficientes MFCC; janela de 20 ms; 40 filtros Mel; frequência mínima de 20 Hz; frequência máxima de 8,000 Hz”, com 250 janelas e classificador com uma camada oculta de tamanho igual à dimensão do vetor de entrada, que atingiu os melhores resultados: AUC-ROC média de 0,7289 e AP média de 0,1938.

Tabela 5 – Dez configurações de MFCC com melhor desempenho, considerando o cenário limpo. A coluna “Modelo” refere-se ao tipo de classificador descrito no texto.

Janelas	**Configuração**	**Modelo**	**AUC-ROC**	**AP**
250	13 MFCC; janela 20 ms; 40 filtros Mel; fmin = 20 Hz; fmax = 8,000 Hz	1	0,7289	0,1938
250	13 MFCC; janela 20 ms; 40 filtros Mel; fmin = 0 Hz; fmax = 8,000 Hz	1	0,7216	0,1850
250	13 MFCC; janela 20 ms; 26 filtros Mel; fmin = 0 Hz; fmax = 8,000 Hz	1	0,7140	0,1770
500	13 MFCC; janela 20 ms; 40 filtros Mel; fmin = 20 Hz; fmax = 8,000 Hz	2	0,7137	0,1762
250	16 MFCC; janela 20 ms; 40 filtros Mel; fmin = 0 Hz; fmax = 8,000 Hz	1	0,7125	0,1783
250	13 MFCC; janela 20 ms; 26 filtros Mel; fmin = 20 Hz; fmax = 8,000 Hz	1	0,7104	0,1740
100	13 MFCC; janela 20 ms; 40 filtros Mel; fmin = 0 Hz; fmax = 8,000 Hz	1	0,7083	0,1711
500	13 MFCC; janela 20 ms; 40 filtros Mel; fmin = 0 Hz; fmax = 8,000 Hz	2	0,7082	0,1713
250	13 MFCC; janela 20 ms; 40 filtros Mel; fmin = 20 Hz; fmax = 8,000 Hz	2	0,7075	0,1692
250	13 MFCC; janela 20 ms; 26 filtros Mel; fmin = 0 Hz; fmax = 8,000 Hz	2	0,7073	0,1691

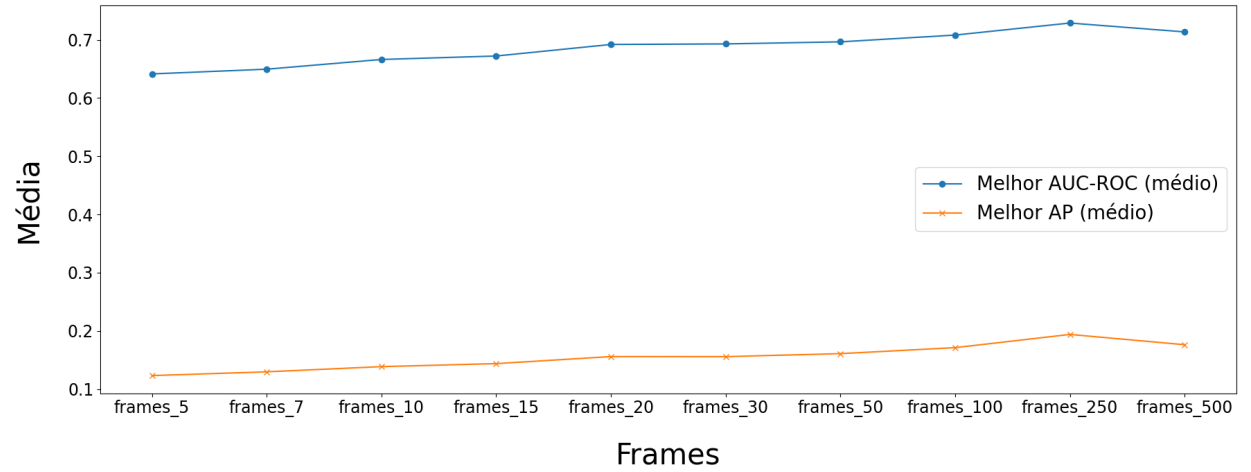
No cenário limpo, a distribuição dos valores de AUC-ROC varia conforme o número de janelas avaliadas. Como mostrado na Figura 21, pode-se observar que as medianas aumentam de forma gradual ao passar de 5 para 30 janelas e continuam subindo em 50 e 100, enquanto a dispersão visivelmente, com caixas mais curtas e os bigodes mais contidos, o que sinaliza resultados mais estáveis entre configurações e repetições. Em 250 janelas obtém-se a mediana mais alta com variabilidade contida. Em 500 janelas a mediana permanece próxima à de 250, com ligeira redução e poucos valores atípicos, indicando um platô entre 250 e 500 janelas.

Figura 21 – Distribuição do AUC-ROC médio por número de janelas no cenário limpo.



Os valores médios de AUC-ROC e AP variam conforme o número de janelas consideradas na representação. Na Figura 22 observa-se que o desempenho é mais baixo nas contagens muito pequenas, entre 5 e 7 janelas, e cresce à medida que o número de janelas aumenta, atingindo o melhor ponto em 250 janelas, com AUC-ROC aproximadamente 0,72 e AP aproximadamente 0,19. Em 100 e em 500 janelas os valores permanecem próximos ao máximo, com leve recuo em 500. Esse comportamento sugere que aumentar o número de janelas amplia a cobertura temporal antes da agregação por média e estabiliza o vetor, de modo que, a partir de 250 janelas, os ganhos tendem a ser marginais.

Figura 22 – Evolução dos melhores valores de AUC-ROC e AP por número de janelas no cenário limpo.



Com base nesses resultados, a configuração “13 MFCC; janela 20 ms; 40 filtros Mel; fmin = 20 Hz; fmax = 8,000 Hz” com 250 janelas e **Modelo 1** foi selecionada como a mais adequada para compor a comparação com modelos baseados em aprendizagem profunda.

4.2 Análise Comparativa com Outras Representações

Após a identificação da melhor configuração de MFCC, foi realizada uma comparação entre MFCC, CLMR e VGGish utilizando o mesmo *pipeline* experimental, os mesmos particionamentos de dados e os mesmos hiperparâmetros de treinamento dos *probes*. Essa comparação não tem como objetivo estabelecer um novo estado da arte em *autotagging*, mas validar, na prática, a integração dos MFCCs ao `mir_ref`. O fato de ser possível comparar diretamente o MFCC integrado ao `mir_ref` com o CLMR e o VGGish, representações produzidas por redes neurais profundas já disponíveis no *framework*, demonstra que a nova abordagem de representação de áudio se encaixa corretamente no fluxo do `mir_ref` e pode ser utilizada como opção ao lado das demais representações.

Para evidenciar a comparação direta entre as representações sob a mesma capacidade do classificador, os resultados foram organizados por modelo. As Tabelas 6–8 apresentam, para cada modelo, quatro colunas de medidas: **AUC-ROC Média**, **AUC-ROC Desvio**, **AP Média**, **AP Desvio**. O desvio-padrão, abreviado como desvio, quantifica a variabilidade entre as 10 execuções independentes; valores menores indicam resultados mais consistentes.

Tabela 6 – Desempenho por representação no **Modelo 0**, considerando o cenário limpo. Valores reportados como média e desvio-padrão para AUC-ROC e AP.

Representação	**AUC-ROC Média**	**AUC-ROC Desvio**	**AP Média**	**AP Desvio**
CLMR	0,8846	0,0005	0,4012	0,0008
VGGish	0,8001	0,0016	0,2698	0,0020
MFCC	0,6864	0,0132	0,1528	0,0101

Tabela 7 – Desempenho por representação no **Modelo 1**, considerando o cenário limpo. Valores reportados como média e desvio-padrão para AUC-ROC e AP.

Representação	**AUC-ROC Média**	**AUC-ROC Desvio**	**AP Média**	**AP Desvio**
CLMR	0,8943	0,0009	0,4167	0,0015
VGGish	0,8662	0,0020	0,3531	0,0031
MFCC	0,7290	0,0059	0,1938	0,0064

Tabela 8 – Desempenho por representação no **Modelo 2**, considerando o cenário limpo. Valores reportados como média e desvio-padrão para AUC-ROC e AP.

Representação	**AUC-ROC Média**	**AUC-ROC Desvio**	**AP Média**	**AP Desvio**
CLMR	0,8950	0,0009	0,4178	0,0028
VGGish	0,8845	0,0008	0,3854	0,0023
MFCC	0,7075	0,0112	0,1692	0,0120

Para facilitar a visualização, os resultados de AUC-ROC e AP por representação e modelo são resumidos em gráficos de barras com médias e desvios-padrão. As Figuras 23 e 24 exibem, respectivamente, os valores para AUC-ROC e AP. Nota-se que as abordagens fundamentadas em aprendizagem profunda, CLMR e VGGish, apresentam desempenho superior em ambas as medidas, com melhor resultado geral para o CLMR no **Modelo 2**. Os desvios-padrão baixos para CLMR e VGGish sugerem alta consistência entre as repetições, enquanto os MFCCs apresentam maior variabilidade.

Figura 23 – AUC-ROC médio e desvio padrão por representação e modelo no cenário limpo.

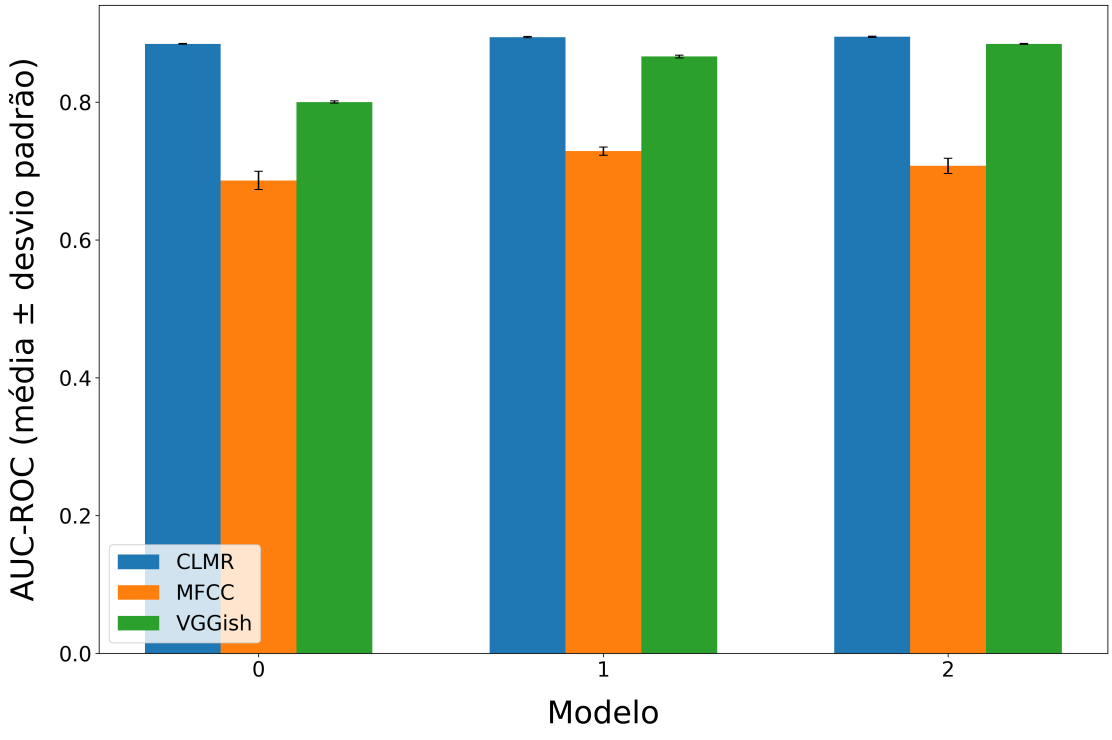
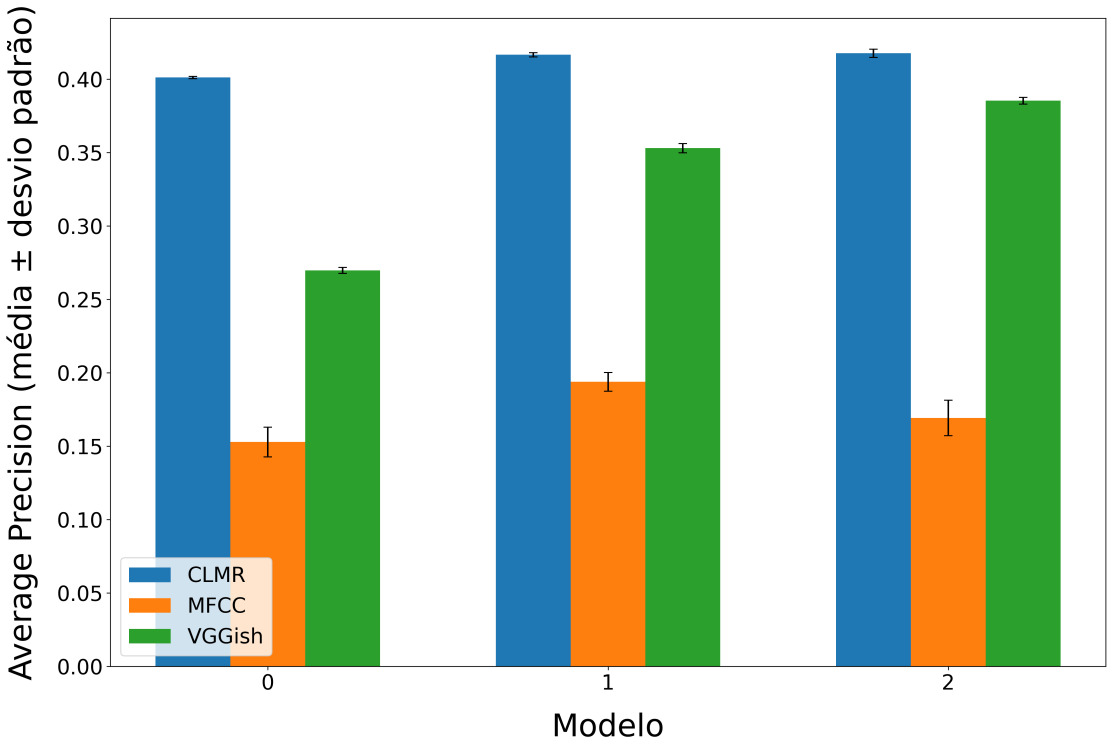


Figura 24 – AP (Precisão Média) e desvio padrão por representação e modelo no cenário limpo.



Em conjunto, os resultados reforçam as vantagens das representações obtidas por aprendizagem profunda, como CLMR e VGGish, para autotagging no cenário limpo e, ao mesmo tempo, evidenciam as limitações dos MFCCs como método de referência, sobretudo na capacidade de capturar nuances relevantes para a tarefa.

5 Conclusão

Este trabalho apresentou a implementação e a integração de uma nova funcionalidade no `mir_ref` para extração, configuração e avaliação de MFCCs em tarefas de classificação musical. A extensão proposta ampliou as possibilidades de experimentação e análise dentro do `mir_ref`, tornando a ferramenta mais flexível e acessível à comunidade de MIR, ao favorecer comparações transparentes e reprodutíveis entre representações clássicas e modernas sob um mesmo protocolo experimental.

No estudo de caso em autotagging multirrótulo, foram adotados o mesmo *pipeline* e os mesmos particionamentos de dados, com classificadores padronizados denominados Modelos 0, 1 e 2 aplicados a todas as representações. O fato da comparação ser realizada de forma direta, sob as mesmas condições experimentais, indica que a integração dos MFCCs ao `mir_ref` foi bem-sucedida. Os resultados indicam desempenho superior das abordagens baseadas em aprendizagem profunda, como CLMR e VGGish, em ambas as medidas AUC-ROC e AP quando comparadas aos MFCCs. Ainda assim, os MFCCs permanecem relevantes como *baseline* interpretável e de baixo custo computacional, especialmente em aplicações

com restrições de recursos. Entre as configurações investigadas de MFCCs, destacou-se a combinação com 13 coeficientes, 40 filtros Mel, fmin = 20 Hz, fmax = 8,000 Hz, janela de 20 ms, avanço de 50% entre janelas, DCT-II com normalização ortonormal e 250 janelas após padronização temporal.

A implementação envolveu a avaliação sistemática de diferentes configurações de extração dos MFCCs e a seleção de parâmetros adequados à tarefa. Para conduzir e comparar os experimentos, foi desenvolvida uma rotina de organização e consolidação de resultados que calcula médias e desvios-padrão ao longo de repetições independentes, o que facilitou a análise comparativa entre MFCCs, CLMR e VGGish e entre os classificadores padronizados Modelos 0, 1 e 2, além de fortalecer a reprodutibilidade dos testes. Ressalta-se que essa rotina foi empregada no contexto deste estudo, enquanto o repositório público original do `mir_ref`, anterior as modificações propostas, mantém sua interface padrão de apresentação de resultados. A versão incrementada com suporte à extração de MFCCs, desenvolvida neste trabalho, encontra-se disponível em https://github.com/JoaoSartoreto/mir_ref. Além do avanço técnico, o processo proporcionou enriquecimento acadêmico e prático ao discente autor do trabalho, favorecendo o desenvolvimento de competências técnicas, científicas e colaborativas.

A integração desta solução reforça o `mir_ref` como plataforma aberta e colaborativa para experimentos em MIR, beneficiando pesquisadores e estudantes interessados na avaliação e comparação de diferentes estratégias de representação de áudio sob condições padronizadas.

5.1 Trabalhos Futuros

Como perspectivas, destacam-se: i) incorporar coeficientes *delta* e *delta-delta* aos MFCCs para capturar variações temporais de curto prazo, prática consolidada em reconhecimento de fala e frequentemente adotada em MIR [63, 64, 21]; ii) aprimorar a rotina automática de organização e consolidação de resultados, visando maior eficiência no gerenciamento de experimentos em larga escala; iii) avaliar cenários adversos com ruídos e distorções, a fim de investigar a robustez das representações e dos classificadores padronizados; iv) explorar estratégias de fusão entre MFCCs e *embeddings* de modelos profundos, aproveitando informações complementares entre as representações [65, 66, 67, 68]. Esses desdobramentos tendem a aprimorar o `mir_ref` e a fortalecer a cultura de pesquisa colaborativa e aberta na área.

Referências

1 BOGDANOV, D. et al. Essentia: An open-source library for sound and music analysis. In: *Proceedings - 21st ACM International Conference on Multimedia*. [S.l.: s.n.], 2013. p. 855–858. Citado 8 vezes nas páginas 8, 13, 14, 17, 23, 24, 32 e 35.

2 MÜLLER, M. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer International Publishing, 2015. ISBN 9783319219455. Disponível em: https://books.google.com.br/books?id=HCI_CgAAQBAJ. Citado 10 vezes nas páginas 8, 12, 13, 14, 15, 16, 17, 18, 19 e 20.

3 ZANGERLE, E. et al. Can microblogs predict music charts? an analysis of the relationship between #nowplaying tweets and music charts. In: *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*. New York City, USA: [s.n.], 2016. p. 365–371. Disponível em: https://wp.nyu.edu/ismir2016/wp-content/uploads/sites/2294/2016/07/039_Paper.pdf. Citado na página 8.

4 BITTNER, R. M. et al. mirdata: Software for reproducible usage of datasets. In: *International Society for Music Information Retrieval Conference*. [s.n.], 2019. Disponível em: https://archives.ismir.net/ismir2019/paper/000009.pdf. Citado 4 vezes nas páginas 8, 9, 29 e 36.

5 HUMPHREY, E.; DURAND, S.; MCFEE, B. Openmic-2018: An open data-set for multiple instrument recognition. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference*. ISMIR, 2018. p. 438–444. Disponível em: https://doi.org/10.5281/zenodo.1492445. Citado na página 8.

6 SOLEYMANI, M. et al. 1000 songs for emotional analysis of music. In: *Proceedings of CrowdMM (ACM Multimedia Workshop)*. [S.l.: s.n.], 2013. Citado na página 8.

7 CHOI, K. et al. Transfer learning for music classification and regression tasks. *CoRR*, abs/1703.09179, 2017. Disponível em: http://arxiv.org/abs/1703.09179. Citado 4 vezes nas páginas 8, 9, 24 e 26.

8 RAFFEL, C. et al. mir_eval: A transparent implementation of common mir metrics. In: *Proceedings - 15th International Society for Music Information Retrieval Conference (ISMIR 2014)*. [S.l.: s.n.], 2014. Citado na página 9.

9 PONS, J.; SERRA, X. musicnn: Pre-trained convolutional neural networks for music audio tagging. *CoRR*, abs/1909.06654, 2019. Disponível em: http://arxiv.org/abs/1909.06654. Citado 2 vezes nas páginas 9 e 13.

10 GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. http://www.deeplearningbook.org. Citado 2 vezes nas páginas 9 e 24.

11 HERSHEY, S. et al. Cnn architectures for large-scale audio classification. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2017. p. 131–135. Citado 3 vezes nas páginas 9, 10 e 13.

12 SPIJKERVET, J.; BURGOYNE, J. A. Contrastive learning of musical representations. *CoRR*, abs/2103.09410, 2021. Disponível em: https://arxiv.org/abs/2103.09410. Citado 5 vezes nas páginas 9, 10, 13, 27 e 28.

13 CRAMER, J. et al. Look, listen, and learn more: Design choices for deep audio embeddings. In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2019. p. 3852–3856. Citado 2 vezes nas páginas 9 e 13.

14 YUAN, R. et al. *MARBLE: Music Audio Representation Benchmark for Universal Evaluation*. 2023. Disponível em: https://arxiv.org/abs/2306.10548. Citado 4 vezes nas páginas 9, 28, 32 e 33.

15 ALONSO-JIMÉNEZ, P.; SERRA, X.; BOGDANOV, D. Music representation learning based on editorial metadata from Discogs. In: *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR)*. Bengaluru, India: [s.n.], 2022. p. 825–833. Disponível em: https://repositori.upf.edu/bitstream/handle/10230/54158/Alonso_ismir_musi.pdf. Citado na página 9.

16 MANCO, I. et al. *Contrastive Audio-Language Learning for Music*. 2022. Disponível em: https://arxiv.org/abs/2208.12208. Citado na página 9.

17 MCFEE, B. et al. Open-source practices for music signal processing research: Recommendations for transparent, sustainable, and reproducible audio research. *IEEE Signal Processing Magazine*, v. 36, p. 128–137, 01 2019. Citado na página 9.

18 DOWNIE, J. et al. The music information retrieval evaluation exchange: Some observations and insights. In: _____. [S.l.]: Springer Berlin Heidelberg, 2010. v. 274, p. 93–115. ISBN 978-3-642-11673-5. Citado 3 vezes nas páginas 9, 32 e 33.

19 CHOPRA, A.; ROY, A.; HERREMANS, D. *MIRFLEX: Music Information Retrieval Feature Library for Extraction*. 2024. Disponível em: https://arxiv.org/abs/2411.00469. Citado 2 vezes nas páginas 9 e 32.

20 PLACHOURAS, C.; ALONSO-JIMéNEZ, P.; BOGDANOV, D. *mir_ref: A Representation Evaluation Framework for Music Information Retrieval Tasks*. 2023. Disponível em: https://arxiv.org/abs/2312.05994. Citado 3 vezes nas páginas 10, 33 e 34.

21 LOGAN, B. Mel frequency cepstral coefficients for music modeling. *Proc. 1st Int. Symposium Music Information Retrieval*, 11 2000. Citado 11 vezes nas páginas 10, 13, 14, 19, 21, 23, 24, 35, 36, 37 e 46.

22 KINSLER, L. E. et al. *Fundamentals of Acoustics*. 4th. ed. [S.l.]: John Wiley & Sons, 2000. Citado na página 11.

23 KINSLER, L. et al. *Fundamentals of Acoustics*. Wiley, 2000. ISBN 9780471847892. Disponível em: https://books.google.com.br/books?id=FecSEAAAQBAJ. Citado na página 11.

24 FLETCHER, N.; ROSSING, T. *The Physics of Musical Instruments*. Springer New York, 2012. (Springer Study Edition). ISBN 9781461229803. Disponível em: https://books.google.com.br/books?id=gvDSBwAAQBAJ. Citado na página 11.

25 OPPENHEIM, A. V.; SCHAFER, R. W.; BUCK, J. R. *Discrete-Time Signal Processing*. 2nd. ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1999. ISBN 0137549202. Citado 8 vezes nas páginas 12, 14, 15, 16, 17, 18, 19 e 20.

26 TZANETAKIS, G.; COOK, P. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, IEEE, v. 10, n. 5, p. 293–302, 2002. Citado 4 vezes nas páginas 12, 24, 29 e 36.

27 MCFEE, B. et al. librosa: Audio and music signal analysis in python. In: *Proceedings of the 14th python in science conference*. [S.l.: s.n.], 2015. p. 18–24. Citado 14 vezes nas páginas 13, 14, 17, 18, 19, 21, 22, 23, 24, 32, 35, 36, 37 e 38.

28 HOHMANN, V. Frequency analysis and synthesis using a gammatone filterbank. *Acta Acustica united with Acustica*, v. 88, p. 433–442, 05 2002. Citado na página 13.

29 ZWICKER, E.; FASTL, H. *Psychoacoustics: Facts and Models*. Springer Berlin Heidelberg, 2013. (Springer Series in Information Sciences). ISBN 9783662095621. Disponível em: https://books.google.com.br/books?id=WLvtCAAAQBAJ. Citado 3 vezes nas páginas 13, 14 e 22.

30 GRIFFIN, D. W.; LIM, J. S. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 32, n. 2, p. 236–243, abr. 1984. Citado na página 17.

31 ALLEN, J. B.; RABINER, L. R. A unified approach to short-time fourier analysis and synthesis. *Proceedings of the IEEE*, v. 65, n. 11, p. 1558–1564, nov. 1977. Citado 3 vezes nas páginas 17, 19 e 20.

32 HAYKIN, S. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1999. (International edition). ISBN 9780132733502. Disponível em: https://books.google.com.br/books?id=bX4pAQAAMAAJ. Citado na página 24.

33 CORTES, C.; VAPNIK, V. Support-vector networks. *Machine Learning*, v. 20, n. 3, p. 273–297, 1995. ISSN 1573-0565. Disponível em: https://doi.org/10.1007/BF00994018. Citado na página 25.

34 DIELEMAN, S.; SCHRAUWEN, B. End-to-end learning for music audio. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2014. p. 6964–6968. Citado na página 26.

35 ALAIN, G.; BENGIO, Y. *Understanding intermediate layers using linear classifier probes*. 2016. Citado na página 28.

36 CHEN, T. et al. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020. Citado na página 28.

37 HE, K. et al. *Momentum Contrast for Unsupervised Visual Representation Learning*. 2020. Disponível em: https://arxiv.org/abs/1911.05722. Citado na página 28.

38 LAW, E. et al. Evaluation of algorithms using games: The case of music tagging. In: *10th International Society for Music Information Retrieval Conference (ISMIR 2009)*. [S.l.: s.n.], 2009. p. 387–392. Citado na página 29.

39 ENGEL, J. et al. Neural audio synthesis of musical notes with wavenet autoencoders. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. [S.l.: s.n.], 2017. p. 1068–1077. Citado na página 29.

40 BOGDANOV, D. et al. The mtg-jamendo dataset for automatic music tagging. In: *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML 2019)*. Long Beach, CA, United States: [s.n.], 2019. Disponível em: http://hdl.handle.net/10230/42015. Citado na página 29.

41 FAWCETT, T. Introduction to roc analysis. *Pattern Recognition Letters*, v. 27, p. 861–874, 06 2006. Citado 2 vezes nas páginas 30 e 31.

42 MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. *Introduction to Information Retrieval*. USA: Cambridge University Press, 2008. ISBN 0521865719. Citado na página 30.

43 BOYD, K.; ENG, K.; PAGE, C. Area under the precision-recall curve: Point estimates and confidence intervals. In: *Machine Learning and Knowledge Discovery in Databases*. [S.l.: s.n.], 2013. v. 8190, p. 451–466. ISBN 978-3-642-38708-1. Citado na página 31.

44 Music Technology Group (UPF). *Essentia Models — Discogs EffNet (feature extractors)*. n.d. https://essentia.upf.edu/models/feature-extractors/discogs-effnet/. Página oficial de distribuição dos pesos. Acessado em: 2025-10-19. Citado na página 34.

45 ALONSO-JIMÉNEZ, P. et al. Tensorflow audio models in Essentia. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2020. Citado na página 34.

46 ALONSO-JIMÉNEZ, P.; SERRA, X.; BOGDANOV, D. Efficient supervised training of audio transformers for music representation learning. In: INTERNATIONAL SOCIETY FOR MUSIC INFORMATION RETRIEVAL. *Proceedings of the 24th International Society for Music Information Retrieval Conference (ISMIR 2023)*. [S.l.], 2023. Citado na página 34.

47 CHANG, S. et al. Neural audio fingerprint for high-specific audio retrieval based on contrastive learning. In: *ICASSP 2021 – IEEE International Conference on Acoustics, Speech and Signal Processing*. [S.l.: s.n.], 2021. p. 3025–3029. Citado na página 34.

48 CHANG, S. et al. *Neural Audio Fingerprint for High-specific Audio Retrieval based on Contrastive Learning*. 2021. Disponível em: https://arxiv.org/abs/2010.11910. Citado na página 34.

49 LI, Y. et al. *MERT: Acoustic Music Understanding Model with Large-Scale Self-supervised Training*. 2024. Disponível em: https://arxiv.org/abs/2306.00107. Citado na página 34.

50 VIMAL, B. et al. Mfcc based audio classification using machine learning. In: *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. [S.l.: s.n.], 2021. p. 1–4. Citado 2 vezes nas páginas 36 e 37.

51 GHOSAL, D. A. et al. Music classification based on mfcc variants and amplitude variation pattern: A hierarchical approach. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, v. 5, p. 131–150, 03 2012. Citado 2 vezes nas páginas 36 e 37.

52 THIRUVENGATANADHAN, R. Music classification using mfcc and svm. *International Research Journal of Engineering and Technology (IRJET)*, v. 5, n. 9, p. 922–924, 2018. Disponível em: https://www.irjet.net/archives/V5/i9/IRJET-V5I9170.pdf. Citado 2 vezes nas páginas 36 e 37.

53 LI, T.; OGIHARA, M.; LI, Q. A comparative study on content-based music genre classification. In: *SIGIR 2003: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 28 - August 1, 2003, Toronto, Canada*. [S.l.]: ACM, 2003. p. 282–289. Citado 2 vezes nas páginas 36 e 37.

54 EYBEN, F. *Real-time Speech and Music Classification by Large Audio Feature Space Extraction*. [S.l.]: Springer Cham, 2016. ISBN 978-3-319-27298-6. Citado 2 vezes nas páginas 36 e 37.

55 RAJESH, B.; BHALKE, D. Automatic genre classification of indian tamil and western music using fractional mfcc. *International Journal of Speech Technology*, v. 19, 09 2016. Citado 2 vezes nas páginas 36 e 37.

56 LI, T.; CHAN, A. Genre classification and the invariance of mfcc features to key and tempo. In: *Advances in Multimedia Modeling*. [S.l.: s.n.], 2011. v. 6523, p. 317–327. ISBN 978-3-642-17831-3. Citado 2 vezes nas páginas 36 e 37.

57 BHALKE, D. G.; RAO, C. B. R.; BORMANE, D. S. Automatic musical instrument classification using fractional fourier transform based- mfcc features and counter propagation neural network. *Journal of Intelligent Information Systems*, v. 46, p. 425 – 446, 2015. Disponível em: https://api.semanticscholar.org/CorpusID:6609123. Citado 2 vezes nas páginas 36 e 37.

58 JENSEN, J. et al. Evaluation of mfcc estimation techniques for music similarity. In: *2006 14th European Signal Processing Conference*. Florence, Italy: [s.n.], 2006. Disponível em: https://vbn.aau.dk/files/5757792/jensen06mfcc.pdf. Citado 2 vezes nas páginas 36 e 37.

59 DAVIS, S. B.; MERMELSTEIN, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In: _____. *Readings in Speech Recognition*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990. p. 65–74. ISBN 1558601244. Citado 2 vezes nas páginas 36 e 37.

60 QIN, Z.; LIU, W.; WAN, T. A bag-of-tones model with mfcc features for musical genre classification. In: *Advanced Data Mining and Applications*. [S.l.]: pringer Berlin Heidelberg, 2013. v. 8346, p. 564–575. ISBN 978-3-642-53913-8. Citado 2 vezes nas páginas 36 e 37.

61 RAWAT, P. et al. A comprehensive study based on mfcc and spectrogram for audio classification. *Journal of Information and Optimization Sciences*, v. 44, p. 1057–1074, 01 2023. Citado 2 vezes nas páginas 36 e 37.

62 MCFEE, B. et al. *librosa.feature.mfcc — librosa 0.10.0 documentation*. 2024. Acesso em: 21 jul. 2025. Disponível em: https://librosa.org/doc/main/generated/librosa.feature.mfcc.html. Citado na página 37.

63 FURUI, S. Speaker-independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 34, n. 1, p. 52–59, 1986. Citado na página 46.

64 YOUNG, S. et al. *The HTK book*. [S.l.]: Cambridge University Engineering Department, 2002. Citado na página 46.

65 SIMONETTA, F.; NTALAMPIRAS, S.; AVANZINI, F. Multimodal music information processing and retrieval: Survey and future challenges. In: *2019 International Workshop on Multilayer Music Representation and Processing (MMRP)*. [S.l.: s.n.], 2019. p. 10–18. Citado na página 46.

66 ORAMAS, S. et al. Multimodal deep learning for music genre classification. *Transactions of the International Society for Music Information Retrieval*, v. 1, p. 4–21, 09 2018. Citado na página 46.

67 ORAMAS, S. et al. Multi-label music genre classification from audio, text and images using deep features. In: *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*. Suzhou, China: [s.n.], 2017. p. 23–30. Disponível em: https://archives.ismir.net/ismir2017/paper/000126.pdf. Citado na página 46.

68 ALFARO-CONTRERAS, M. et al. Late multimodal fusion for image and audio music transcription. *Expert Systems with Applications*, v. 216, p. 119491, 2023. ISSN 0957-4174. Disponível em: https://www.sciencedirect.com/science/article/pii/S0957417422025106. Citado na página 46.

Apêndices

APÊNDICE A – Tutorial de uso da integração de MFCCs no `mir_ref`

1 Integração ao *pipeline* do `mir_ref`: Tutorial de Uso

A implementação foi planejada para compatibilidade com as rotinas do `mir_ref`, permitindo que os MFCCs sejam usados em qualquer tarefa suportada, sem necessidade de adaptações extras. Para um novo usuário, o funcionamento segue o padrão do *framework*:

1. **Escolha do *dataset*:** Os arquivos de áudio devem estar acessíveis e organizados conforme o formato e o padrão de entrada definidos pelo *framework*, por exemplo, com *datasets* integrados à `mirdata`.

2. **Configuração dos parâmetros:** No arquivo de configuração `.yml` do `mir_ref`, define-se o tipo de extração de características como `mfcc` e ajustam-se os parâmetros desejados. Um exemplo de configuração é:

```
features:
  #- vggish-audioset
  #- clmr-v2
  #- mert-v1-95m-6
  - mfcc
feature_parameters:
  mfcc:
    n_mfcc: 13
    n_mels: 40
    fmin: 20
    fmax: 8000
    dct_type: 2
    norm: ortho
    window_size_ms: 20
    max_frames: 250
```

3. **Extração:** A extração é realizada com:

```
python run.py extract -c seu_arquivo_config.yml
```

4. **Treinamento:** Após a extração das características, realiza-se o treinamento com:

```
python run.py train -c seu_arquivo_config.yml
```

5. **Avaliação:** Em seguida, realiza-se a avaliação com:

```
python run.py evaluate -c seu_arquivo_config.yml
```

6. **Registro e comparação de resultados:** A saída do comando de avaliação é exibida no terminal; para registrá-la em arquivo texto, pode-se redirecionar a saída:

```
python run.py evaluate -c seu_arquivo_config.yml > resultado.txt
```

O código-fonte completo está disponível no repositório GitHub do projeto: https://github.com/JoaoSartoreto/mir_ref.