CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS TIMÓTEO

Wylton Leone França

ALGORITMO DE RECONHECIMENTO DE ÁUDIO PARA A PROTEÇÃO DE DIREITOS AUTORAIS

Timóteo

2024

Wylton Leone França

ALGORITMO DE RECONHECIMENTO DE ÁUDIO PARA A PROTEÇÃO DE DIREITOS AUTORAIS

Monografia apresentada à Coordenação de Engenharia de Computação do Campus Timóteo do Centro Federal de Educação Tecnológica de Minas Gerais para obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Dr. Elder de Oliveira Rodrigues

Timóteo

2024

Wylton Leone França

Algoritmo de Reconhecimento de Áudio para a Proteção de Direitos Autorais

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais, campus Timóteo, como requisito parcial para obtenção do título de Engenheiro de Computação.

Trabalho aprovado. Timóteo, 12 de fevereiro de 2025:

Prof. Dr. Elder de Oliveira Rodrigues
Orientador

Prof. Me. Adilson Mendes Ricardo
Professor Convidado

Prof. Me. Odilon Corrêa da Silva
Professor Convidado

Timóteo
2024

Agradecimentos

Agradeço primeiramente a Deus e aos meus pais, que me deram forças e condições para continuar sempre estudando e me dedicando. Sem o apoio incondicional e os ensinamentos transmitidos ao longo da minha jornada, essa conquista não teria sido possível.

Expresso minha sincera gratidão ao meu orientador, Elder, por acreditar no meu trabalho. Seu apoio foi fundamental para o desenvolvimento deste projeto, assim como para outros trabalhos realizados ao longo da minha trajetória acadêmica.

Por fim, não poderia deixar de mencionar o matemático Leonhard Euler, cujas contribuições atemporais para a matemática embasaram este trabalho e continuam sendo uma inspiração para pesquisadores e estudantes em diversas áreas do conhecimento.

Resumo

A proteção de direitos autorais no ambiente digital apresenta constantes desafios, principalmente na detecção e monitoramento de conteúdos audiovisuais compartilhados em plataformas online. A crescente disseminação de materiais protegidos exige soluções tecnológicas para garantir a integridade dos direitos dos criadores. Este estudo propõe o desenvolvimento de um algoritmo de reconhecimento de áudio fundamentado na técnica de impressão digital (*fingerprint*), com o objetivo de viabilizar a identificação e a proteção de conteúdos sujeitos a direitos autorais. O algoritmo emprega a Transformada de Fourier (FT) para extrair características únicas dos sinais sonoros, gerando representações hash que possibilitam a identificação de padrões acústicos específicos. A metodologia adotada envolveu a implementação do algoritmo e a avaliação da sua acurácia em uma base de dados diversificada, composta por diferentes gêneros musicais. Foram realizados testes utilizando amostras que variam de 1 a 5 segundos. Os resultados obtidos demonstram que a abordagem proposta é eficaz na identificação de trechos de áudio com duração superior a 1.8 segundos. Além disso, identificou-se que o gênero musical influencia na quantidade de hashes gerados em decorrência da quantidade de picos encontrados no espectrograma, impactando a precisão do reconhecimento. O trabalho contribui para o aprimoramento de técnicas de proteção de direitos autorais e apresenta aplicabilidade em sistemas de monitoramento de conteúdo em larga escala. Além disso, o estudo se mostrou uma síntese de algoritmos de processamento de áudio, fornecendo suporte para pesquisas futuras na área de análise de sinais.

Palavras-chave: reconhecimento de áudio, impressão digital, transformada de fourier, direitos autorais.

Abstract

Copyright protection in the digital environment presents constant challenges, especially in the detection and monitoring of audiovisual content shared on online platforms. The increasing dissemination of protected materials requires technological solutions to ensure the integrity of creators' rights. This study proposes the development of an audio recognition algorithm based on the *fingerprinting* technique, aiming to enable the identification and protection of content subject to copyright. The algorithm employs the Fourier Transform (FT) to extract unique features from audio signals, generating hash representations that allow the identification of specific acoustic patterns. The adopted methodology involved implementing the algorithm and evaluating its accuracy in a diverse database composed of different musical genres. Tests were conducted using samples ranging from 1 to 5 seconds. The obtained results demonstrate that the proposed approach is effective in identifying audio segments longer than 1.8 seconds. Furthermore, it was identified that the musical genre influences the number of generated hashes due to the number of peaks found in the spectrogram, impacting the accuracy of recognition. This work contributes to the improvement of copyright protection techniques and has applicability in large-scale content monitoring systems. Additionally, the study proved to be a synthesis of audio processing algorithms, providing support for future research in the field of signal analysis.

Keywords: audio recognition, fingerprinting, fourier transform, copyright.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ADC	Conversor Analógico-Digital
API	*Application Programming Interface*
CD	*Compact Disc*
CPU	*Central Processing Unit*
dB	Decibéis
DFT	*Discrete Fourier Transform*
DSP	*Digital Signal Processing* (Processamento Digital de Sinais)
FFT	*Fast Fourier Transform*
GB	Gigabyte
GPU	*Graphics Processing Unit*
Hz	Hertz
kB	Kilobyte
MP3	*MPEG-1 Audio Layer 3*
RAM	*Random Access Memory*
STFT	*Short-Time Fourier Transform*

Sumário

1 Introdução

A proteção dos direitos autorais na era digital tem se tornado um tema cada vez mais relevante, especialmente com a evolução tecnológica que trouxe desafios significativos à indústria fonográfica. A transição do mercado, marcada pela propagação da pirataria nos anos 90 e 2000, criou um cenário complexo que exige uma reavaliação das práticas e legislações vigentes, especialmente a Lei de Direitos Autorais (Lei 9.610/1998) (ARLOTTA, 2017).

Por outro lado, a tecnologia desempenha um papel cada vez mais importante na proteção dos direitos autorais. A técnica de impressão digital de áudio, também conhecida como *fingerprint*, exemplifica como a inovação pode ser utilizada para este propósito. Esta solução tem sido amplamente adotada por plataformas como o YouTube, que, por meio do sistema *Content ID* (GOOGLE, 2024), monitora e identifica automaticamente o uso indevido de obras protegidas por direitos autorais, garantindo um controle sobre o uso do conteúdo.

A técnica de *fingerprint* permite a análise de arquivos de áudio, extraindo características únicas que possibilitam sua identificação em outros arquivos (CANO et al., 2005). Estas características permitem que uma faixa de áudio seja reconhecida, mesmo em meio a grandes volumes de dados ou quando submetida a pequenas modificações. Na Seção 2.4, serão introduzidos os detalhes técnicos desta técnica, explicando os princípios matemáticos e computacionais que sustentam seu funcionamento.

1.1 Problema

O aumento da disseminação de conteúdo digital e a facilidade de compartilhamento pela internet tornam a violação de direitos autorais um problema frequente. As plataformas de compartilhamento de mídia precisam se adequar às legislações vigentes e, atualmente, enfrentam desafios na identificação e controle de conteúdos que violam os direitos dos criadores. A ausência de técnicas acessíveis de reconhecimento de áudio dificulta a proteção de tais direitos, resultando em prejuízos financeiros e desestímulos à criação artística (ARLOTTA, 2017).

Diante deste cenário, esta monografia busca responder às seguintes perguntas: quais são as principais dificuldades e limitações dos algoritmos atuais de reconhecimento de áudio? E de que maneira um algoritmo de reconhecimento de áudio, utilizando a técnica de *fingerprint*, pode ser desenvolvido para identificar, de forma eficiente, violações de direitos autorais em plataformas de compartilhamento de mídia?

1.2 Justificativa

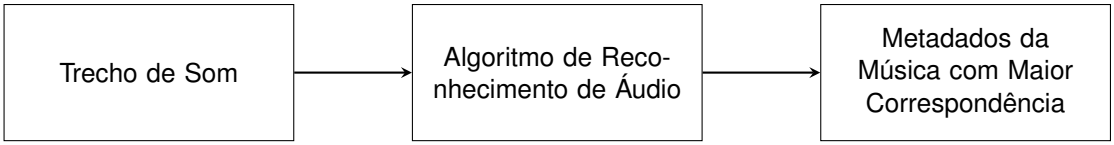
Esta monografia justifica-se pela crescente necessidade de desenvolver métodos eficientes e acessíveis para a proteção dos direitos autorais na era digital. Além de abordar o

processo de implementação de um algoritmo de reconhecimento de áudio, este trabalho oferece uma síntese dos principais conceitos de processamento de áudio, evidenciando como essas tecnologias podem ser aplicadas de forma prática.

1.3 Objetivos

Este trabalho de conclusão de curso tem como objetivo principal o desenvolvimento de um algoritmo de reconhecimento de áudio voltado para a proteção de direitos autorais. O algoritmo será baseado na técnica de *fingerprint*, que permitirá a identificação de características únicas em arquivos sonoros, facilitando a detecção de possíveis violações. O resultado esperado é ilustrado na Figura 1, onde um trecho de som é processado pelo algoritmo de reconhecimento de áudio para indexar os metadados da música com maior correspondência no banco de dados.

Figura 1 – Fluxograma Esperado do Processo de Reconhecimento de Áudio



Fonte: Elaborada pelo autor

Os objetivos específicos deste trabalho incluem:

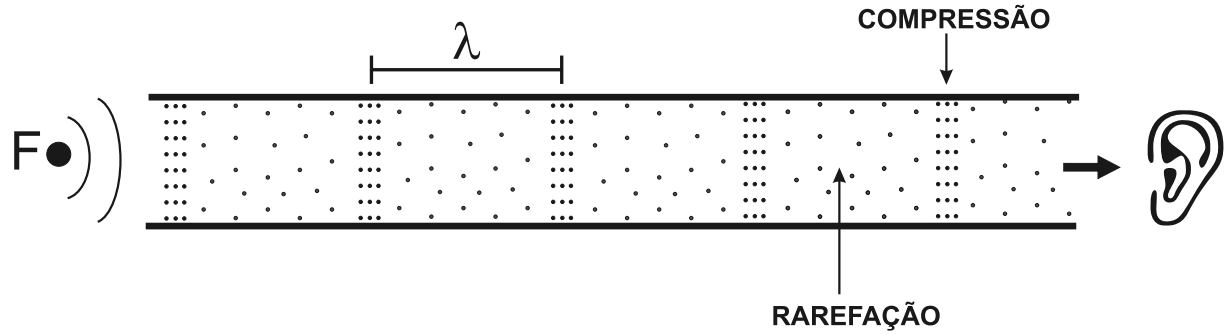
1. Implementar os algoritmos para a geração e comparação de *fingerprints* de áudio.
2. Escolher um banco de dados apropriado e armazenar as *fingerprints* geradas a partir de amostras de áudio de uma base de dados.
3. Realizar testes para avaliar a precisão e o desempenho do algoritmo na identificação de amostras de áudio extraídas de músicas.

2 Fundamentação teórica

2.1 Som

O som é uma sensação auditiva percebida pelo nosso ouvido. Sempre que escutamos um som, é porque um objeto está vibrando. Essas vibrações criam ondas sonoras, as quais são perturbações mecânicas que se propagam por meios elásticos, como sólidos, líquidos ou gases (BORGES; RODRIGUES, 2017). Conforme ilustrado na Figura 2, a vibração do objeto é transmitida de molécula a molécula pelo ar até alcançar nossos ouvidos, permitindo-nos ouvir o som.

Figura 2 – Fonte sonora produzindo uma onda acústica em um tubo de ar.



Fonte: (BORGES; RODRIGUES, 2017)

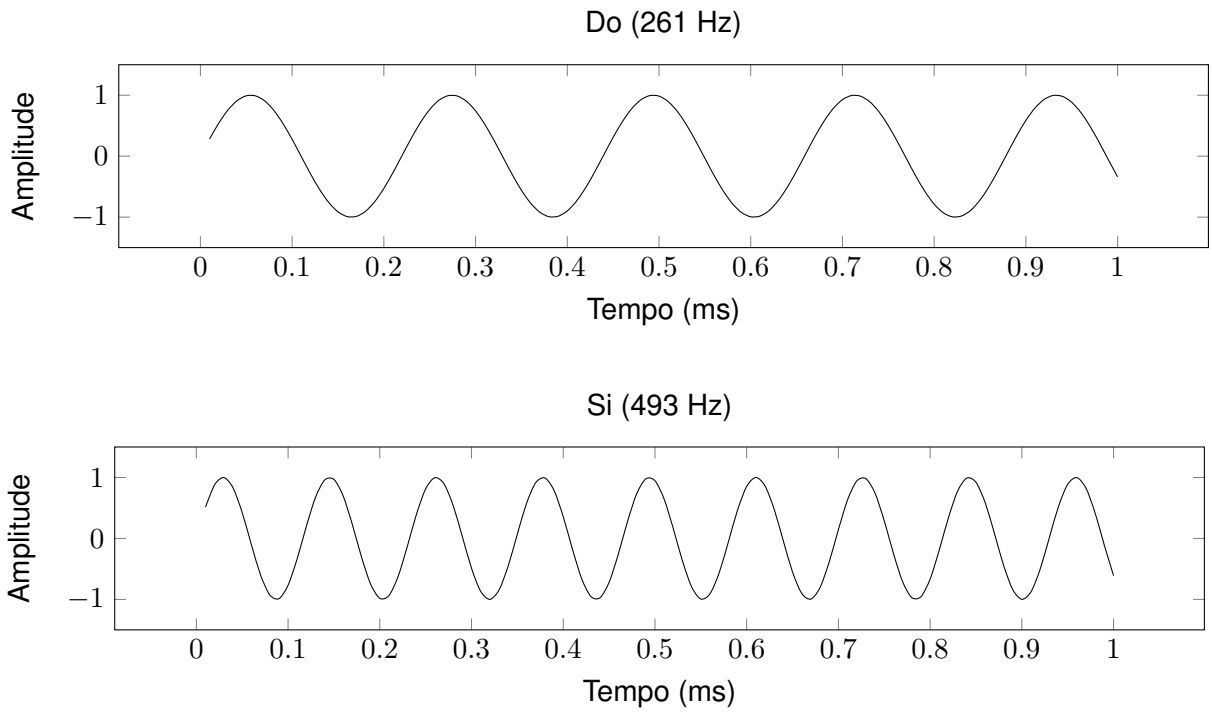
2.1.1 Características das Ondas Sonoras

As ondas sonoras possuem características que determinam como o som é percebido:

1. **Frequência**: A frequência é a medida do número de ciclos completos de vibração que ocorrem em um segundo, sendo expressa em Hertz (Hz). Em termos de percepção auditiva, sons com frequências mais altas são percebidos como agudos, enquanto os de frequências mais baixas são percebidos como graves. A variação na frequência é o que permite ao ouvido humano distinguir diferentes notas musicais e tonalidades.

   Como demonstrado na Figura 3, notas musicais com diferentes frequências, como Dó (261 Hz) e Si (493 Hz), possuem um número distinto de oscilações num mesmo intervalo de tempo. Isso significa que a nota Si, por ter uma frequência mais alta, realiza mais oscilações por segundo do que a nota musical Dó.

Figura 3 – Ondas senoidais para as notas musicais Dó e Si.



Fonte: elaborada pelo autor

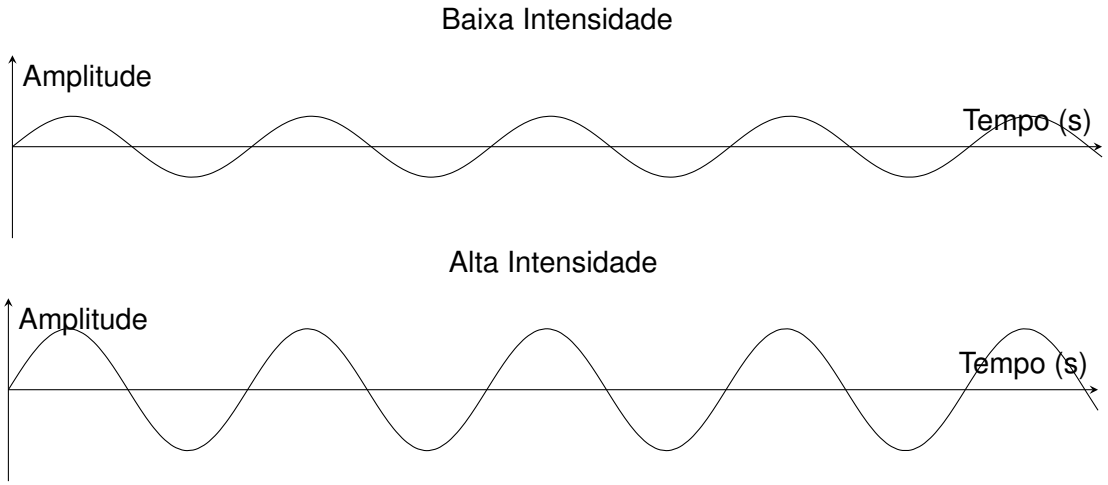
A faixa de audição humana típica varia de aproximadamente 20 Hz a 20.000 Hz. Sons fora dessa faixa não são perceptíveis ao ouvido humano, sendo que frequências abaixo de 20 Hz são chamadas de infrassons e acima de 20.000 Hz são chamadas de ultrassons (BORGES; RODRIGUES, 2017).

2. **Intensidade**: A intensidade sonora é a medida da magnitude da variação da pressão sonora, percebida pelo ouvinte como o volume do som, sendo expressa em decibéis (dB). A intensidade está diretamente associada à amplitude das ondas sonoras. Quanto maior a amplitude, maior será a pressão sonora gerada e, consequentemente, maior será a intensidade percebida pelo ouvido humano.

   Essa característica da onda também está intimamente relacionada à quantidade de energia transportada pela onda sonora ao longo de sua propagação. Sons com alta intensidade podem causar desconforto auditivo ou até mesmo danos ao aparelho auditivo, enquanto sons de baixa intensidade podem ser fracos a ponto de se tornarem inaudíveis (BORGES; RODRIGUES, 2017).

   A Figura 4 apresenta uma ilustração de ondas sonoras com diferentes intensidades. As ondas de baixa intensidade possuem menor amplitude, resultando em um som mais suave e menos invasivo. Por outro lado, as ondas de alta intensidade apresentam uma amplitude maior, gerando um som mais forte e, portanto, mais facilmente perceptível pelo ouvido humano.

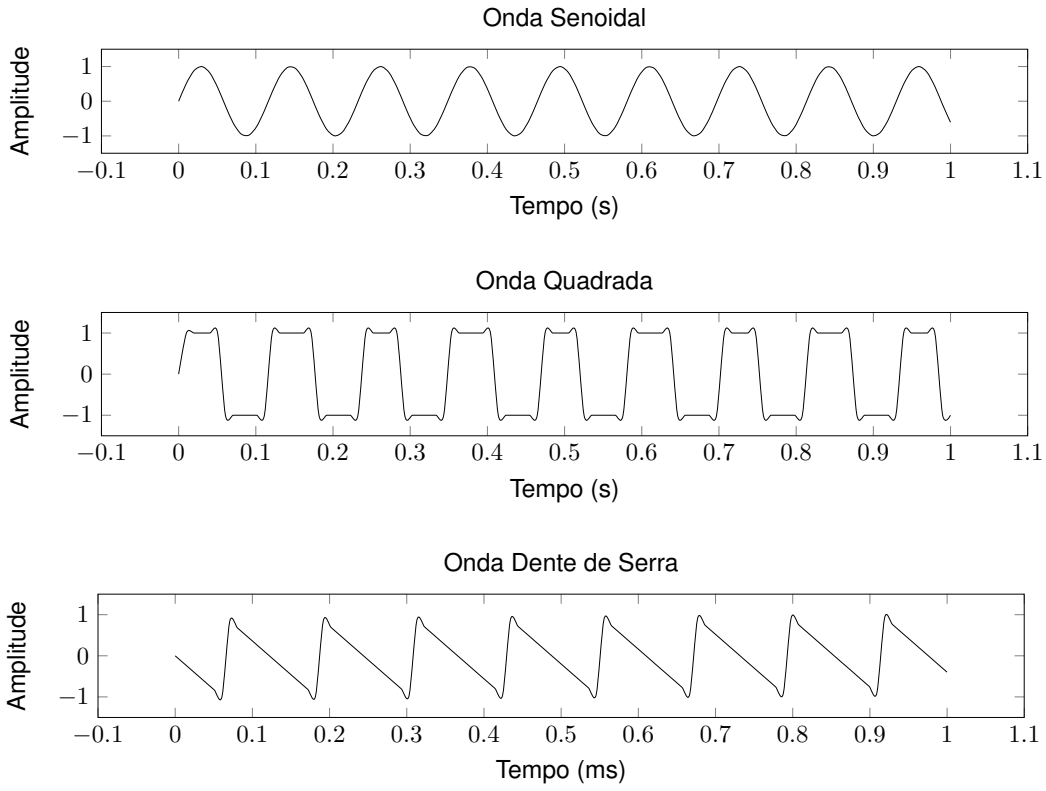
Figura 4 – Ondas sonoras com diferentes intensidades.



Fonte: elaborada pelo autor

3. **Forma de Onda**: A forma de onda descreve o formato da onda sonora e caracteriza o timbre, que consiste nas características que permitem diferenciar diferentes fontes sonoras. O timbre é o que permite identificar se um som vem de um violino, um piano ou uma guitarra, ainda que estejam tocando a mesma nota musical na mesma intensidade (BORGES; RODRIGUES, 2017).

Figura 5 – Formas de onda senoidal, quadrada e dente de serra.



Fonte: elaborada pelo autor

Essa diversidade de formas de onda é explorada para criar identidades sonoras únicas em músicas. A Figura 5 ilustra diferentes tipos de formas de onda, como senoidal, quadrada e dente de serra, exemplificando como cada uma delas apresenta uma configuração distinta, apesar de possuírem a mesma frequência e amplitude.

2.2 Sinal Analógico

Um sinal pode ser definido como qualquer quantidade física que varia em função do tempo, espaço ou de outra variável independente. Matematicamente, um sinal é descrito como uma função que depende de uma ou mais variáveis independentes (PROAKIS; MANOLAKIS, 1996). No contexto do som, essa variável é geralmente o tempo, e o sinal sonoro corresponde à intensidade da onda sonora ao longo do tempo.

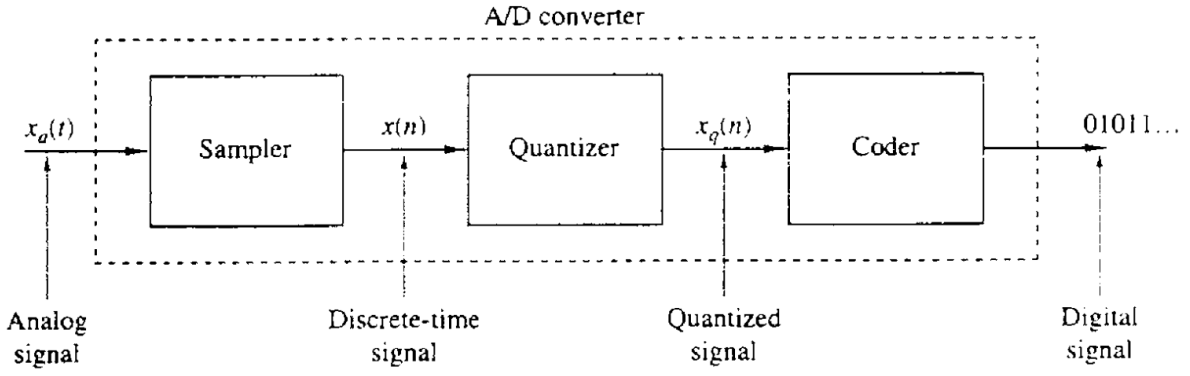
Sinais de interesse prático, como os de áudio e vídeo, são tipicamente analógicos. Isto significa que eles variam de forma contínua no tempo e assumem valores em um intervalo contínuo. Em outras palavras, a amplitude de um sinal analógico pode assumir infinitos valores possíveis num determinado intervalo. Estes sinais podem ser processados por sistemas analógicos, mas não podem ser manipulados diretamente por sistemas digitais.

2.3 Digitalização

Para que sinais analógicos possam ser processados por meios digitais, como em um computador, é necessário realizar a conversão destes sinais para o formato digital. Isto envolve transformar o sinal contínuo em uma sequência finita de números, com precisão finita, que podem ser interpretados por dispositivos digitais.

Este procedimento é conhecido como conversão analógico-digital, e os dispositivos responsáveis por esta tarefa são chamados de conversores analógico-digital (ADCs) (PROAKIS; MANOLAKIS, 1996). O processo de conversão é realizado em três etapas, conforme ilustrado na Figura 6.

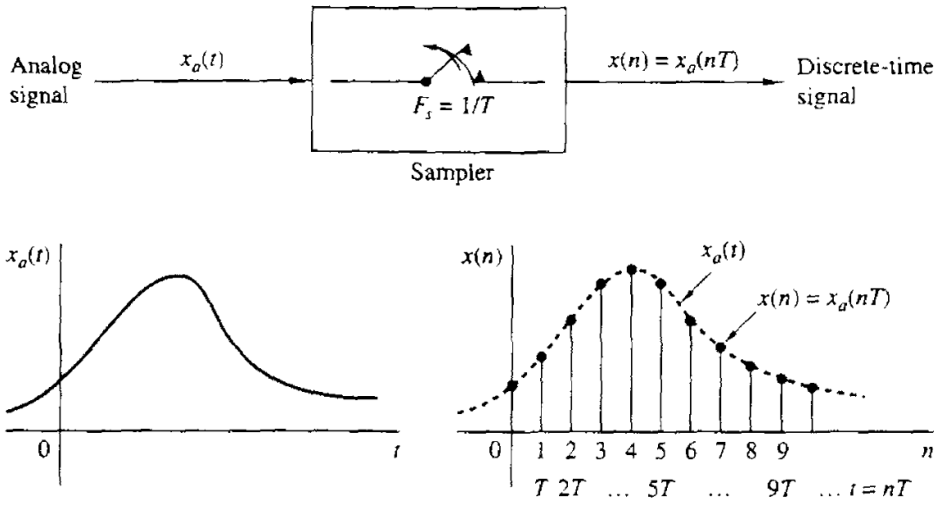
Figura 6 – Partes básicas de um conversor analógico-digital (A/D).



Fonte: (PROAKIS; MANOLAKIS, 1996)

1. **Amostragem**: Na amostragem, o sinal analógico contínuo no tempo é medido em intervalos de tempo discretos, como exemplificado na Figura 7. A frequência de amostragem deve ser, no mínimo, duas vezes a maior frequência presente no sinal analógico, conforme estabelecido pelo teorema de Nyquist (PROAKIS; MANOLAKIS, 1996).

Figura 7 – Amostragem periódica de um sinal analógico.



Fonte: (PROAKIS; MANOLAKIS, 1996)

Para um áudio de alta fidelidade, a frequência de amostragem comum é de 44,1 kHz, o que permite captar frequências de até 22,05 kHz, cobrindo a faixa de audição humana. A amostragem abaixo desta frequência pode causar um efeito chamado *aliasing*, onde diferentes sinais se tornam indistinguíveis após a amostragem.

2. **Quantização**: Após a amostragem, os valores das amostras ainda são contínuos e podem assumir qualquer valor em um intervalo. A quantização envolve a aproximação desses valores contínuos para um conjunto finito de valores discretos (PROAKIS; MANOLAKIS, 1996), resultando em uma perda de precisão chamada erro de quantização.

   Este processo é necessário em decorrência dos computadores digitais só manipularem valores numéricos discretos. No caso do áudio, a quantização determina a resolução do som digital, comumente representada por 16 bits por amostra em CDs de áudio, permitindo 65.536 níveis de amplitude diferentes.

3. **Codificação**: Na codificação, os valores quantizados são convertidos em uma sequência binária, ou seja, uma sequência de bits que pode ser manipulada e armazenada por sistemas digitais (PROAKIS; MANOLAKIS, 1996). Cada valor quantizado é representado por um código binário, com o número de bits determinando a precisão da representação.

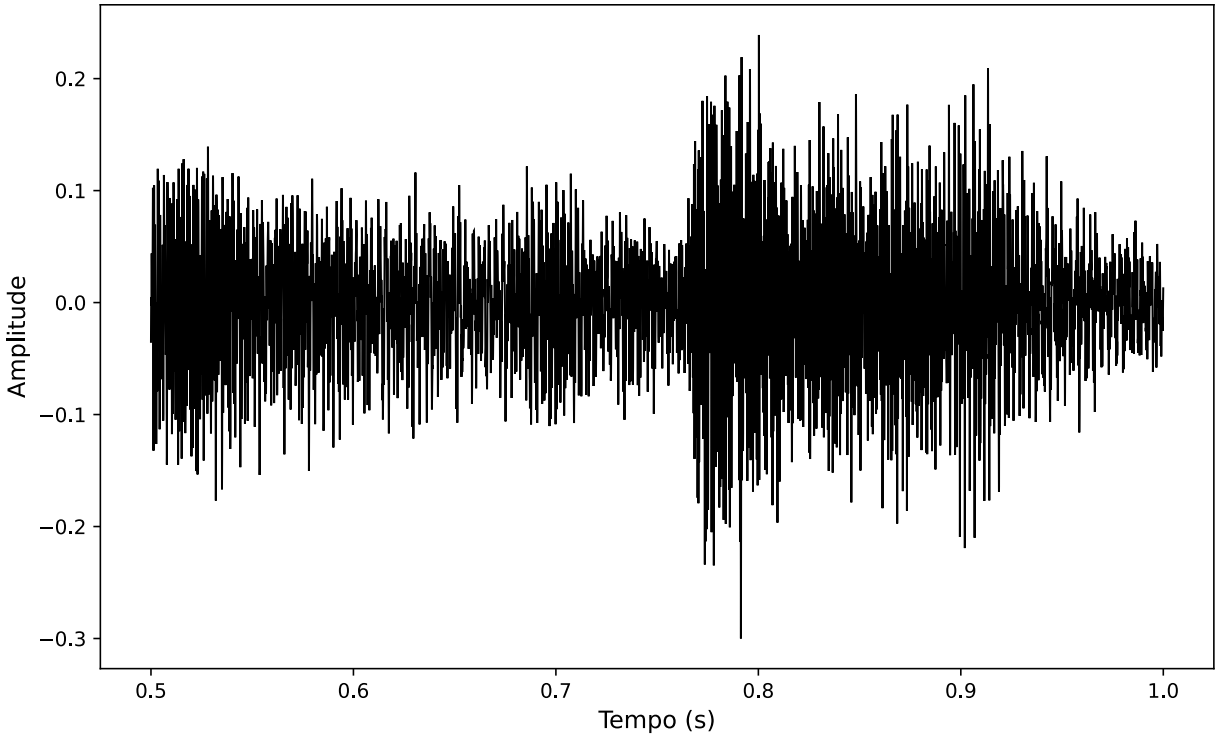
Uma vez que o sinal sonoro foi digitalizado, podemos aplicar algoritmos para processar as amostras obtidas. O processamento do sinal envolve a aplicação de técnicas matemáticas e computacionais para extrair, modificar ou analisar informações contidas no sinal. O proces-

samento pode incluir filtragem, compressão, e transformação do sinal para facilitar a análise subsequente.

2.4 Impressão Digital

Um pequeno trecho de música contém a contribuição de milhares de fontes de som, como voz humana e instrumentos. Estas contribuições resultam em um sinal, conforme a Figura 8, sendo a soma de várias oscilações audíveis em diferentes frequências, intensidades e formas. Se conseguirmos identificar as frequências que melhor representam essas oscilações, juntamente com suas respectivas magnitudes, podemos criar uma impressão digital do áudio, conhecida como *fingerprint* (CANO et al., 2005).

Figura 8 – Representação Temporal de um Trecho de Música.



Fonte: Elaborada pelo autor.

A frequência de uma onda sonora está diretamente relacionada ao número de ciclos que a onda completa em um determinado período. Para sinais simples, é possível iterar sobre os dados e inferir as frequências presentes em um intervalo do sinal. No entanto, esse método torna-se impraticável quando aplicado a trechos mais complexos, como uma música. Para facilitar essa análise, são utilizadas técnicas como as transformadas, que decompõem o sinal em componentes de frequência, facilitando a identificação das características únicas do áudio.

2.5 Transformadas Integrais

Uma transformada integral é uma operação linear, definida pela equação 2.1. A função resultante $F(s)$ é comumente denominada imagem da função $f(t)$. Cada tipo de transformada é caracterizado por um núcleo $K(t, s)$ e um intervalo de integração $[a, b]$, que definem a forma exata da transformação (DEBNATH; BHATTA, 2007).

$$F(s) = \int_a^b f(t)\, K(t, s)\, dt \tag{2.1}$$

Existem diversos tipos de transformadas. A Tabela 1 apresenta uma comparação entre alguns dos principais tipos. Essas transformadas fornecem métodos distintos para interpretar e manipular funções, convertendo a função $f(t)$ em uma nova representação, muitas vezes mais conveniente para análise de sinais, solução de equações diferenciais e outras aplicações.

Tabela 1 – Comparação entre as transformadas de Fourier, Laplace e Wavelet

	Núcleo	**Intervalo de Integração**
Transformada de Fourier	$e^{-i\omega t}$	$-\infty < t < \infty$
Transformada de Laplace	e^{-st}	$0 \leq t < \infty$
Transformada Wavelet	$\frac{1}{\sqrt{\lvert a\rvert}}\psi\left(\frac{t-b}{a}\right)$	$-\infty < t < \infty$

Fonte: Elaborada pelo autor.

2.5.1 Transformada de Fourier

A Transformada de Fourier, definida na Equação 2.2, é um tipo de Transformada Integral. O kernel desta transformada é composto por uma exponencial complexa, que representa componentes oscilatórias, conforme a fórmula de Euler (Equação 2.3). Assim, ao aplicar a Transformada de Fourier, realizamos uma decomposição da função $f(t)$ em uma combinação linear de componentes oscilatórias, cada uma associada a uma frequência ω (OPPENHEIM; WILLSKY; NAWAB, 1996).

$$F(\omega) = \int_{-\infty}^{\infty} f(t)\, e^{-i\omega t}\, dt \tag{2.2}$$

$$e^{-i\omega t} = \cos(\omega t) - i \sin(\omega t) \tag{2.3}$$

Isto permite identificar a contribuição de cada frequência ω na composição da função $f(t)$. Ao realizar a decomposição, a transformada converte a função do domínio do tempo contínuo para o domínio da frequência contínuo (OSGOOD, 2007), demonstrando como diferentes componentes oscilatórias, associadas a frequências específicas, se combinam para formar a função. Desta forma, é possível determinar as frequências que estão presentes em uma função e suas respectivas magnitudes.

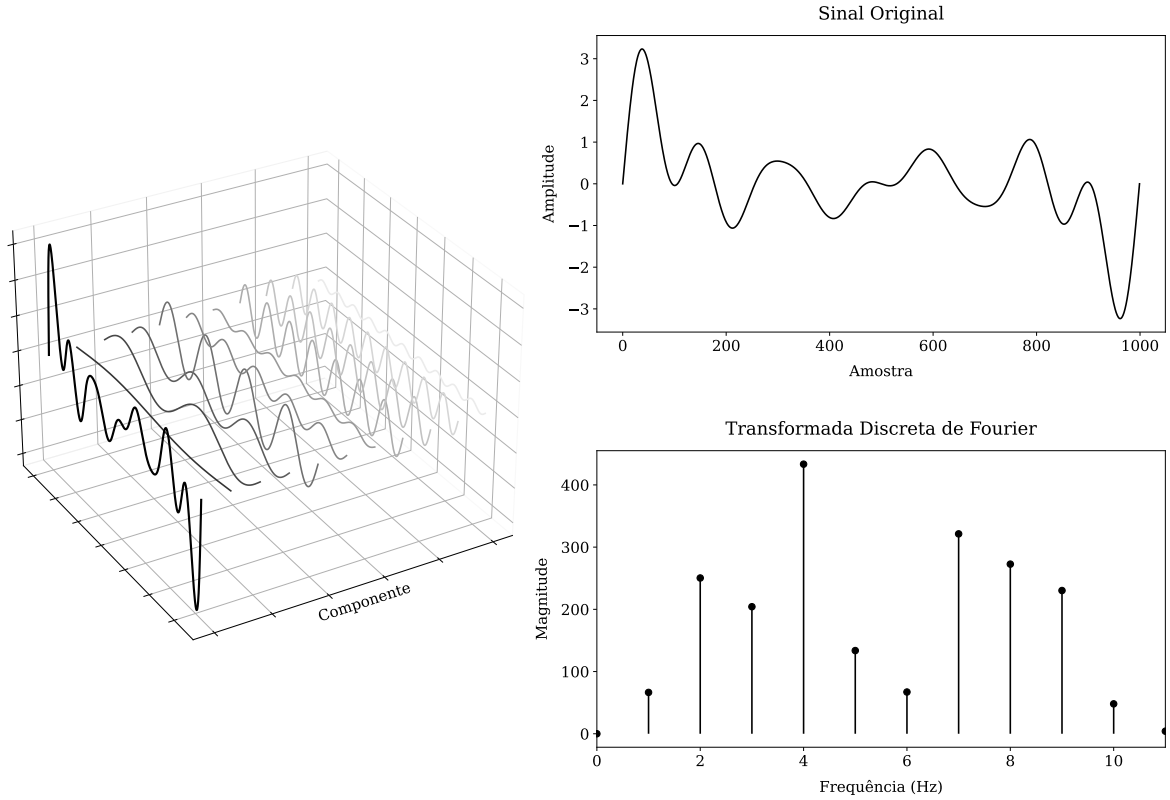
2.5.2 Transformada Discreta de Fourier

A Transformada Discreta de Fourier (DFT) é uma versão amostrada e de duração finita da Transformada de Fourier. A DFT é aplicada a uma sequência de valores discretos no tempo e gera uma sequência discreta de valores no domínio da frequência (OPPENHEIM; WILLSKY; NAWAB, 1996). A entrada $x[n]$ representa o sinal amostrado no tempo, enquanto $X[k]$ é sua correspondente representação no domínio da frequência, amostrada em pontos igualmente espaçados. A equação 2.4 apresenta a definição da DFT.

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j\frac{2\pi}{N}kn}, \quad \text{onde} \quad k = 0, 1, \ldots, N-1 \tag{2.4}$$

A Figura 9 ilustra um sinal sonoro amostrado e suas componentes de frequência obtidas por meio da Transformada Discreta de Fourier. O sinal original é representado em termos de suas amostras. Ao aplicar a DFT, obtemos uma sequência de valores discretos que correspondem às frequências presentes no sinal e suas respectivas magnitudes (LYONS, 2011).

Figura 9 – Análise de Componentes de Frequência via Transformada Discreta de Fourier.



Fonte: Elaborada pelo autor.

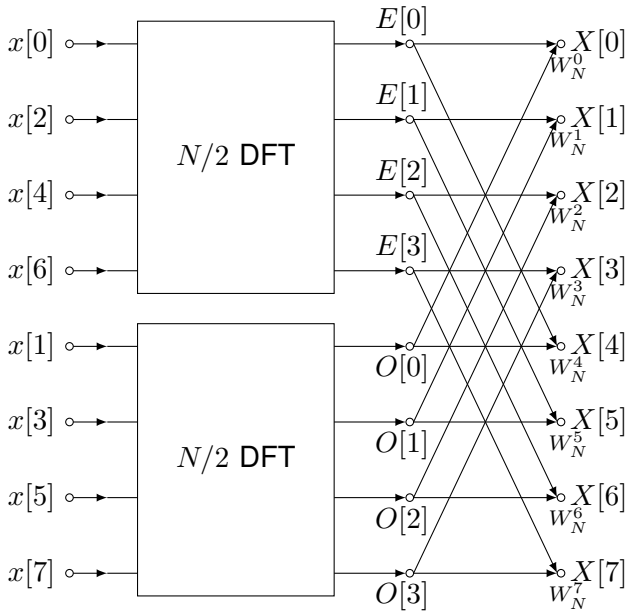
2.5.3 Transformada Rápida de Fourier (FFT)

A Transformada Rápida de Fourier (FFT) é um algoritmo eficiente para calcular a Transformada Discreta de Fourier (DFT), reduzindo o número de operações matemáticas envolvidas no processo. Em comparação com a computação direta da DFT, a FFT minimiza o número de multiplicações necessárias, tornando-a uma ferramenta essencial para o processamento de sinais digitais.

A FFT organiza os cálculos da DFT em uma estrutura otimizada, baseada em uma decomposição hierárquica dos dados. O algoritmo mais comum é a radix-2, que divide os cálculos em estágios sucessivos, reduzindo gradativamente a complexidade computacional (PROAKIS; MANOLAKIS, 1996).

Nesse método, cada estágio do cálculo envolve operações chamadas *butterflies*, que consistem em uma combinação de somas e multiplicações complexas. Como ilustrado na Figura 10, o número de operações decresce a cada estágio, resultando em um total de $O(N \log N)$ operações, em contraste com $O(N^2)$ da DFT convencional.

Figura 10 – Exemplificação do algoritmo FFT radix-2



Fonte: Elaborada pelo autor.

2.6 Banco de Dados

Os bancos de dados são sistemas projetados para armazenar e gerenciar grandes volumes de informações de maneira estruturada e eficiente. Um banco de dados pode ser definido como uma coleção de dados relacionados que possuem um significado implícito, sendo utilizados para diversas finalidades, desde armazenamento de informações textuais até conteúdos multimídia, como imagens, vídeos e áudios (ELMASRI; NAVATHE, 2005).

2.6.1 Banco de Dados Multimídia e Indexação de Áudio

Os bancos de dados multimídia desempenham um papel essencial na organização e recuperação de informações baseadas em elementos visuais e sonoros. A recuperação baseada em conteúdo permite a busca por informações a partir de características extraídas diretamente dos arquivos multimídia, como padrões visuais em imagens ou elementos acústicos em áudios (ELMASRI; NAVATHE, 2005).

No contexto específico de arquivos de áudio, técnicas avançadas de processamento de sinal, como transformadas discretas, podem ser aplicadas para extrair e indexar características únicas dos sons armazenados. Essas características incluem frequência, intensidade, dispersão e pureza, permitindo a identificação de padrões específicos, como a voz de uma pessoa ou trechos musicais.

2.7 Interface de Programação de Aplicações

Uma API (Interface de Programação de Aplicações) é uma especificação que define como diferentes softwares podem interagir entre si. Ela funciona como um contrato entre sistemas, estabelecendo um conjunto de regras e métodos que padronizam essa comunicação. As APIs são amplamente utilizadas para permitir a integração entre aplicações, facilitando o desenvolvimento entre diferentes componentes (HIGGINBOTHAM, 2015).

Um exemplo comum de API é a funcionalidade de caixa de diálogo de confirmação presente nos navegadores. Ao fornecer uma mensagem para essa API, o navegador exibe a informação em uma janela pop-up, oferecendo opções para confirmar ou cancelar a ação (HIGGINBOTHAM, 2015). Dessa forma, as APIs abstraem a complexidade do sistema, permitindo que os desenvolvedores utilizem funcionalidades sem precisar compreender seus detalhes internos.

3 Trabalhos relacionados

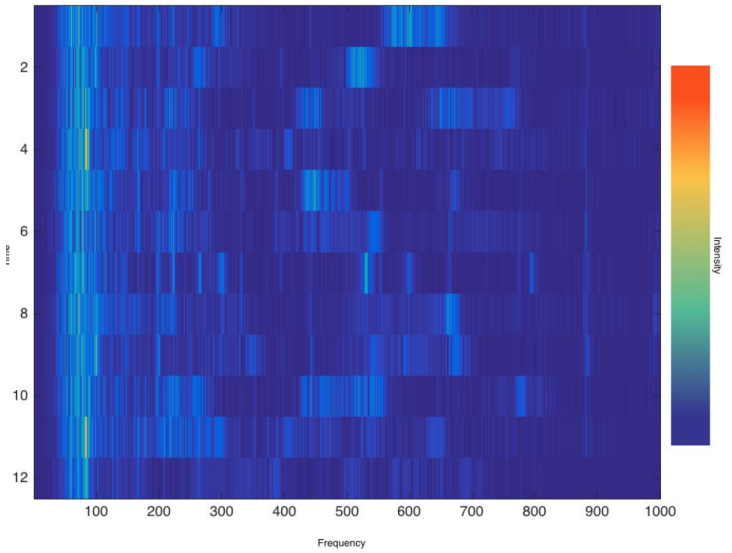
Neste capítulo, serão apresentados os trabalhos que foram encontrados e analisados durante a revisão bibliográfica e que contribuíram para o desenvolvimento desta monografia.

3.1 Um algoritmo de identificação de músicas que utiliza a Análise de Fourier para sinais de áudio

DeChicchis (2016) propôs um sistema de identificação de músicas que utiliza a Transformada de Fourier para a análise de sinais de áudio. O trabalho aborda o processo completo de extração de frequências-chave de uma música, criando uma sequência de frequências predominantes e convertendo-as em um vetor de hash-tempo.

A abordagem é centrada na análise de espectrogramas, onde a Transformada Rápida de Fourier (FFT), uma técnica eficiente para computar a Transformada Discreta de Fourier (DFT), é aplicada para transformar sinais de áudio em um formato que facilita a identificação musical. A Figura 11 ilustra um exemplo de espectrograma gerado a partir de um sinal de áudio, utilizado para destacar as frequências predominantes no processo de identificação.

Figura 11 – Exemplo de um espectrograma de um sinal de áudio.



Fonte: (DECHICCHIS, 2016)

O autor realiza a revisão bibliográfica introduzindo os conceitos sobre a Transformada de Fourier contínua e discreta, a aplicação da transformada na análise de sinais de áudio, espectrograma, extração de frequências mais importantes, criação de hash, armazenamento e identificação de músicas. O algoritmo de identificação de músicas é definido em 6 etapas.

Segundo (DECHICCHIS, 2016, p.2):

> In particular, the music identification algorithm consists of the following parts:
>
> 1. Analyzing audio signals using the Fourier Transform.
> 2. “Sequencing” music by identifying key frequencies.
> 3. Calculating a unique Hash-Time vector for each song.
> 4. Assembling these Hash-Time vector files into a database.
> 5. Running steps 1 through 3 on a sample audio signal.
> 6. Matching the sample audio Hash-Time vector to the correct song from the database.

O trabalho demonstrou que é possível construir um algoritmo de identificação de músicas através da análise de sinais de áudio utilizando a Análise de Fourier. Apesar de o algoritmo de identificação musical ter apresentado resultados satisfatórios, apenas 657 músicas foram incluídas na base de dados de vetores Hash-Tempo, e a eficácia do algoritmo com uma base de dados mais realista contendo milhares de músicas ainda não foi testada, abrindo possibilidade para novos trabalhos.

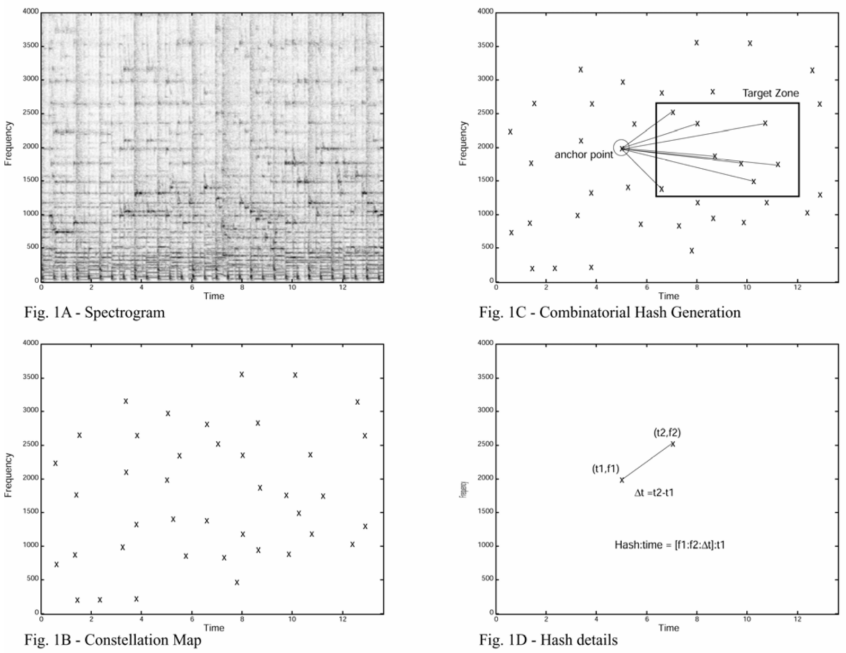
3.2 Um algoritmo de busca de áudio de nível industrial

O trabalho de (WANG, 2003) apresenta um algoritmo de identificação de áudio desenvolvido pela Shazam Entertainment, projetado para ser robusto contra ruídos e distorções, eficiente em termos de processamento e escalável para grandes bases de dados. Este algoritmo é capaz de identificar rapidamente trechos curtos de música capturados por um microfone de celular, mesmo na presença de ruídos de fundo e compressão de codecs de voz, como ocorre em chamadas telefônicas.

O autor descreve o processo tradicional de identificação de música, que começa com a extração de hashes de frequência dos arquivos de áudio. As características sonoras da amostra desconhecida são comparadas com um grande banco de dados de *fingerprints* pré-armazenadas para encontrar correspondências e identificar a música. Além disso, (WANG, 2003) propõe um método baseado em picos do espectrograma para aumentar a precisão da identificação, mesmo em ambientes ruidosos e com distorções.

Um ponto no espectrograma, representado pelo tempo e pela frequência, é considerado um pico candidato, se apresentar uma energia significativamente maior que a de seus vizinhos em uma região ao redor desse ponto. Os picos são selecionados com base em sua densidade, garantindo uma cobertura uniforme da faixa de tempo-frequência do arquivo de áudio. A Figura 12 ilustra o processo de seleção de frequências de um espectrograma usando a técnica de constelações.

Figura 12 – Exemplo de seleção de frequências de um espectrograma



Fonte: (WANG, 2003)

Em seguida, foi retratado o desenvolvimento de um método que reduz o tempo de indexação da base de dados. Este método utiliza a combinação de pares de pontos de tempo-frequência para formar hashes. Pontos específicos são selecionados e emparelhados com outros dentro de uma zona alvo. Esses hashes são compactados em inteiros de 32 bits, e cada um é associado a um deslocamento temporal a partir do início do arquivo. Essa abordagem resulta na aceleração no processo de busca, mesmo em bases de dados contendo milhares de faixas musicais.

Os resultados apresentados por (WANG, 2003) confirmam a eficácia dos métodos implementados. Para um banco de dados com 20 mil faixas, o tempo de busca variou entre 5 e 500 milissegundos. Em casos de áudio de "qualidade de rádio", o tempo de identificação foi inferior a 10 milissegundos. Embora o algoritmo ocasionalmente produzisse falsos positivos, foi observado que essas situações se deviam ao fato de que o sistema captava amostras sonoras similares à faixa correta ou possíveis plágios.

3.3 Identificação e classificação de acordes musicais aplicando a Transformada de Fourier

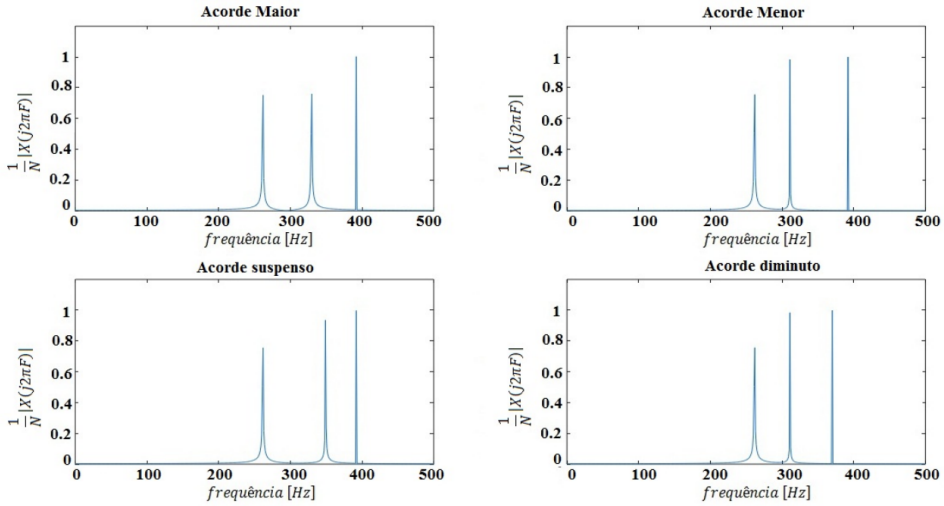
(JESUS, 2019) propõe a identificação e classificação de acordes musicais por meio de um computador utilizando a Transformada de Fourier. O autor começa com uma revisão bibliográfica, abordando conceitos fundamentais de som, séries de Fourier e a Transformada de Fourier, tanto na sua forma contínua quanto discreta. O desenvolvimento do trabalho foi dividido em quatro etapas principais.

Segundo (JESUS, 2019, p.43):

> A implementação prática deste trabalho possui quatro etapas, conforme descrições a seguir. O algoritmo desenvolvido a partir destas etapas foi intitulado Chord Test in Frequency (Teste do Acorde em Frequência), apresentado no Apêndice (A.1).
>
> • Coleta de dados: Realização da gravação dos acordes musicais (executados em um teclado musical);
> • Importação dos dados: Transferência e importação dos áudios gravados no Octave;
> • Processamento dos dados: Cálculo da Transformada de Fourier dos acordes coletados e aplicação da Equação da Escala Temperada para obter a relação musical entre as frequências presentes no sinal;
> • Classificação dos acordes: classificar os acordes com base nos resultados anteriores.

A Figura 13 apresenta a Transformada de Fourier aplicada a alguns acordes musicais, evidenciando como as frequências são decompostas e fornecem as informações necessárias para a análise e classificação dos acordes.

Figura 13 – Exemplo da transformada aplicada em alguns acordes.



Fonte: (JESUS, 2019)

Por fim, o autor apresenta os resultados. Foram avaliados 58 acordes, destes 52 acordes foram devidamente identificados pelo computador e 6 acordes não foram reconhecidos. (JESUS, 2019) constatou que a não identificação ocorre em decorrência do limiar de corte e da ausência de componentes de frequência. Após regravar os acordes com a intensidade adequada, os acordes foram devidamente reconhecidos.

4 Procedimentos metodológicos

A presente pesquisa é exploratória em seus objetivos, pois busca investigar e compreender as principais dificuldades e limitações dos algoritmos atuais de reconhecimento de áudio. É aplicada, uma vez que visa ao desenvolvimento de um método prático, capaz de ser utilizado em cenários reais para identificar possíveis violações de direitos autorais em amostras de áudio. A pesquisa é experimental em seus procedimentos técnicos, envolvendo a implementação e a avaliação de um algoritmo de reconhecimento de áudio (WAZLAWICK, 2009).

4.1 Materiais

Os materiais utilizados nesta pesquisa, incluindo tanto o hardware e as ferramentas de software utilizadas para a programação dos algoritmos, são descritos a seguir:

1. O desenvolvimento e os testes iniciais dos algoritmos serão realizados em um MacBook M1 Pro, com 10 núcleos de CPU e 16GB de RAM.
2. O ambiente de desenvolvimento contará com a utilização do software CLion para a programação dos algoritmos. A linguagem de programação C++ será empregada na implementação e testes do algoritmo.

4.2 Métodos

Os métodos adotados em cada fase do desenvolvimento do sistema, desde a escolha dos algoritmos até a integração final e os testes de desempenho, são detalhados a seguir.

1. **Selecionar e implementar algoritmos para gerar *fingerprints*:**
   a) Implementar os algoritmos de *fingerprint*, baseados na Transformada de Fourier, mais adequados para o reconhecimento de áudio.
2. **Desenvolver API para a entrada de áudio e a persistência e recuperação no banco de dados:**
   a) Criar uma interface que permita a entrada dos arquivos de áudio.
   b) Integrar a API aos algoritmos de *fingerprint* para automatizar o processamento e a um banco de dados para armazenar e recuperar as impressões digitais.
3. **Catalogar amostras de áudio e gerar *fingerprints*:**
   a) Selecionar uma ou mais bases de dados de músicas.
   b) Extrair amostras de áudio dos arquivos selecionados nas bases de dados.

c) Enviar cada amostra de áudio para a API a fim de gerar e armazenar as *fingerprints*.

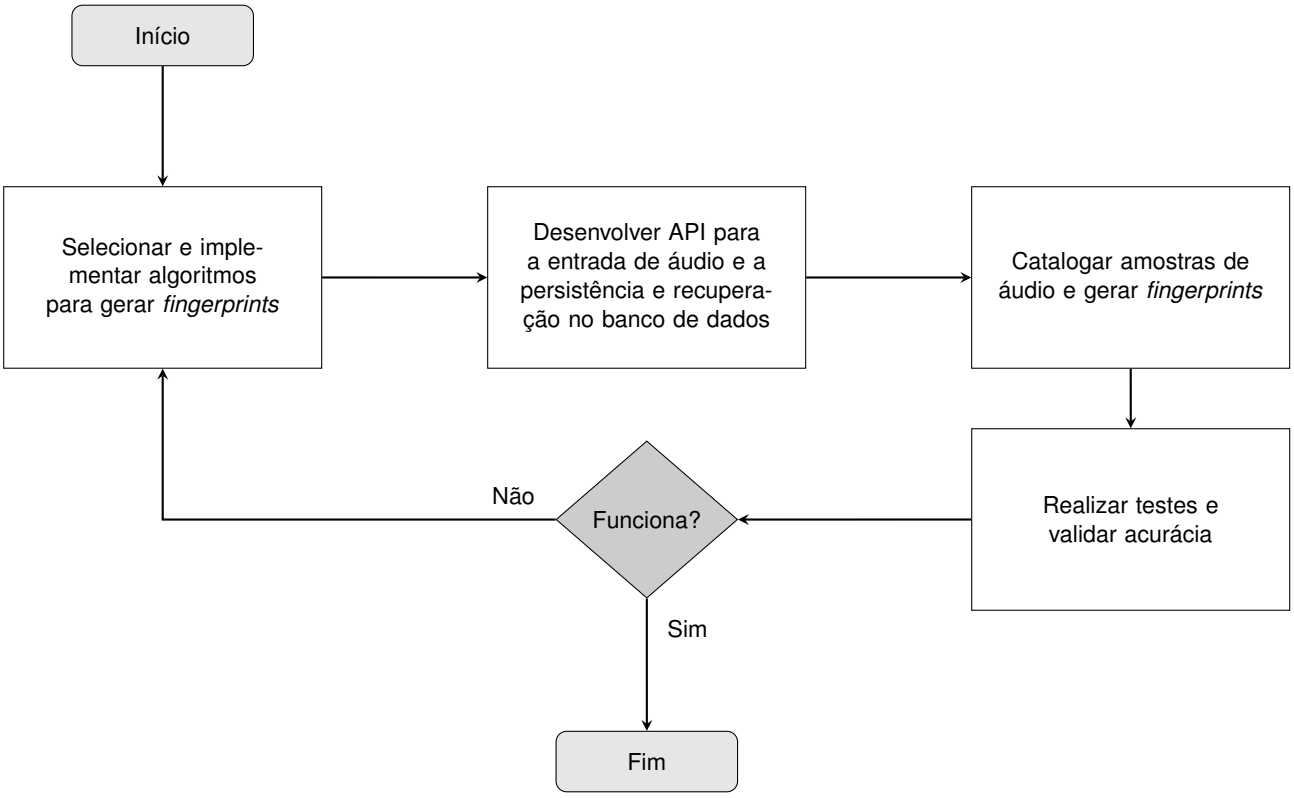
4. **Realizar testes e validar acurácia:**

a) Testar os algoritmos e o banco de dados utilizando um conjunto de amostras de áudio e verificar a acurácia na identificação das músicas catalogadas.

b) Realizar ajustes no algoritmo, caso os resultados não sejam satisfatórios.

A Figura 14 apresenta o fluxograma metodológico do projeto, contendo as quatro etapas para o desenvolvimento do algoritmo de reconhecimento de áudio. Estas etapas incluem desde a implementação de algoritmos até a avaliação final, definindo cada fase do processo de forma sequencial, com a possibilidade de ajustes em etapas anteriores.

Figura 14 – Fluxograma metodológico do projeto



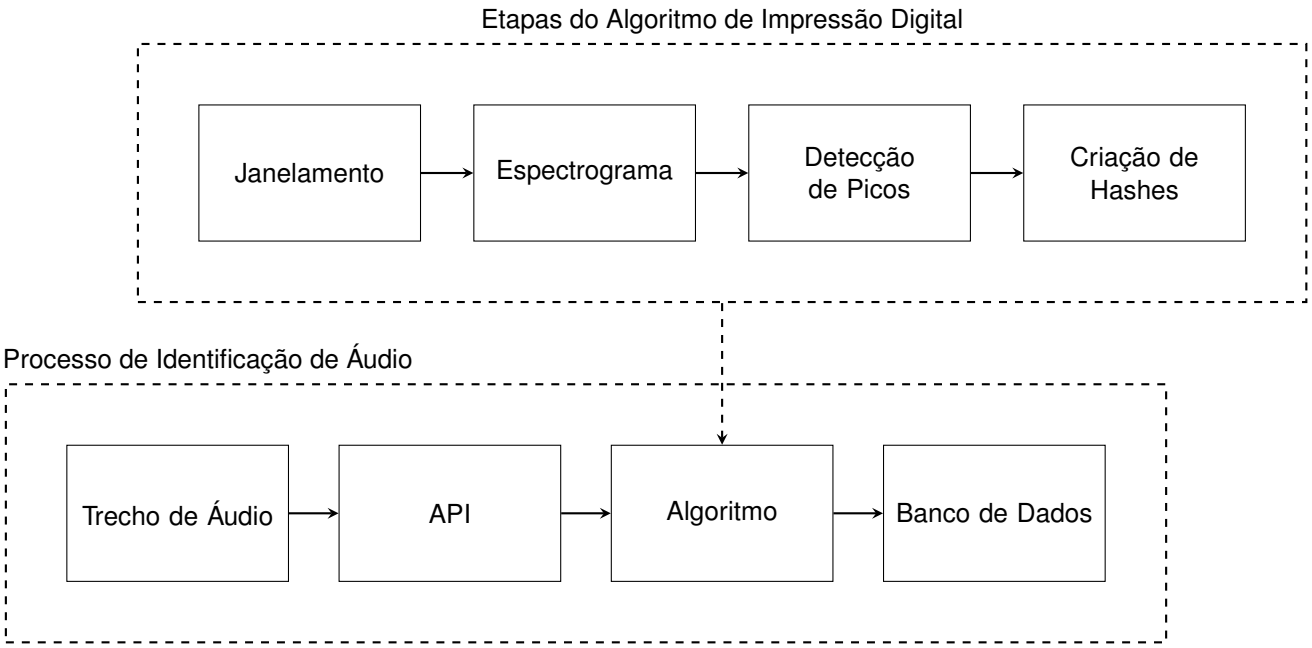
Fonte: elaborada pelo autor

5 Algoritmo de Impressão Digital para o Reconhecimento de Áudio

Neste capítulo, são apresentadas as etapas do desenvolvimento do algoritmo de reconhecimento de áudio, desde o processamento inicial até a validação de sua acurácia. O processo inicia-se com o janelamento do sinal de áudio, seguido pela geração do espectrograma, a detecção de picos e a criação de hashes.

Concluída a implementação, o algoritmo foi aplicado a um conjunto de músicas, cujos hashes foram extraídos e armazenados em um banco de dados. Posteriormente, foram conduzidos testes para avaliar a precisão do sistema na identificação das amostras previamente catalogadas. A Figura 15 ilustra a interconexão dessas etapas.

Figura 15 – Fluxograma do Desenvolvimento



Fonte: Elaborada pelo autor

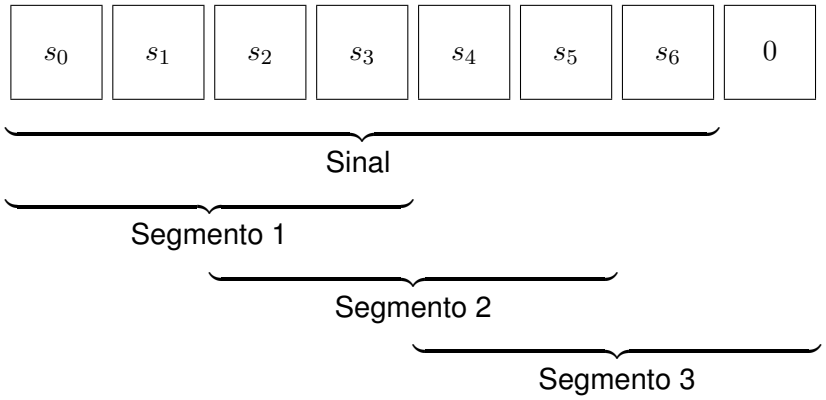
Cada trecho de música foi submetido à API, que decodifica a entrada e encaminha o sinal amostrado para o algoritmo de reconhecimento. Os hashes gerados são armazenados no banco de dados para posterior consulta e validação, utilizado a rota de persistência da API. Para os testes de identificação, foi empregado a rota de busca. Essa rota retorna as informações sobre as músicas que possuem o maior número de hashes correspondentes ao trecho submetido, ordenadas de forma decrescente pelo número de correspondências.

5.1 Janelamento

O primeiro passo no desenvolvimento do algoritmo foi a segmentação do sinal de áudio, um procedimento conhecido como janelamento. Esse processo preserva a informação temporal das frequências, garantindo que a análise espectral represente a evolução do sinal ao longo do tempo. Caso a Transformada de Fourier fosse aplicada diretamente ao sinal completo, as informações de variação temporal seriam perdidas, prejudicando a precisão na identificação sonora.

O janelamento envolve duas etapas: a divisão do sinal em segmentos com sobreposição e a aplicação de uma função de janela a cada segmento. Na primeira etapa, o sinal foi dividido em intervalos fixos de 4096 amostras, utilizando uma sobreposição de 50% entre segmentos consecutivos, conforme ilustrado na Figura 16. A sobreposição assegura uma continuidade temporal nos dados, reduzindo perdas nas transições entre os segmentos.

Figura 16 – Segmentação do Sinal com Sobreposição de 50%



Fonte: Elaborada pelo autor.

O tamanho de 4096 amostras foi escolhido por ser uma potência de 2, um requisito essencial para a aplicação eficiente do algoritmo FFT (Fast Fourier Transform) nas etapas subsequentes. Esse valor equivale a aproximadamente 90 milissegundos de áudio, considerando que o sinal foi amostrado a uma taxa de 44.100 Hz. Além disso, essa escolha otimiza o armazenamento e o processamento das informações, permitindo a criação de impressões digitais utilizando apenas 32 bits, o que reduz a demanda computacional sem comprometer a precisão da identificação sonora. Essa questão será discutida na Seção 5.4.

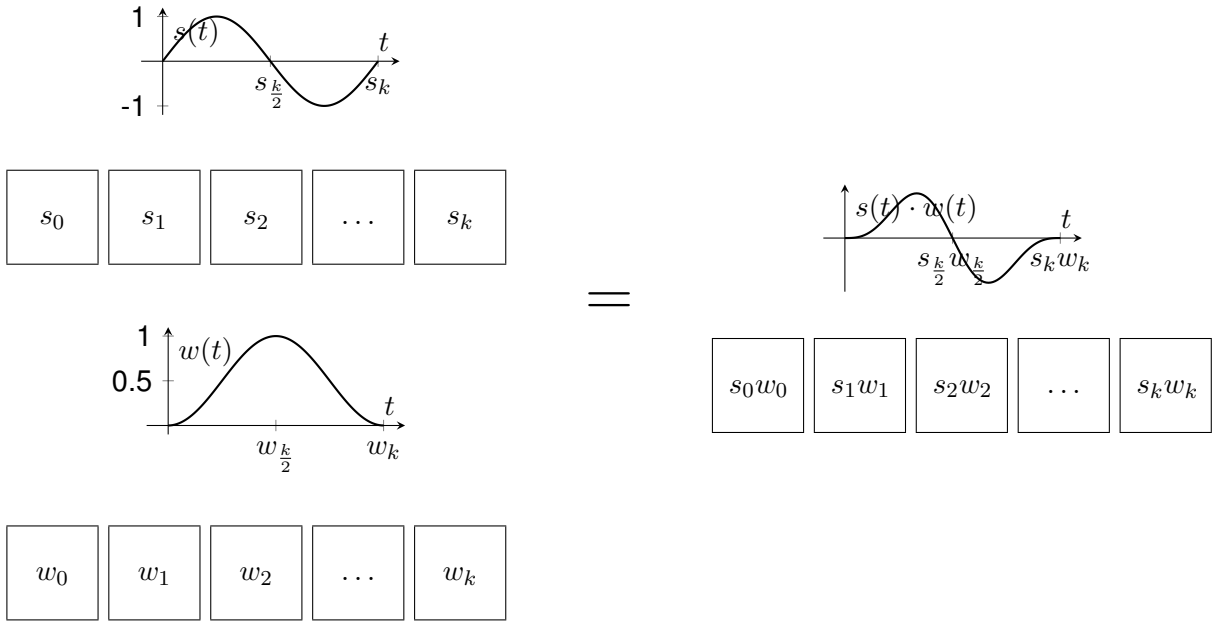
Em casos onde um segmento de áudio fosse menor que 4096 amostras, o restante foi preenchido com zeros, uma técnica conhecida como zero-padding. Esse procedimento assegura que todos os segmentos tenham o mesmo tamanho, garantindo a uniformidade no processamento. Como consequência, cada segmento gerado possuirá 2048 bins de frequência, que representam os componentes espectrais extraídos a partir da Transformada de Fourier.

Após a segmentação, cada trecho foi multiplicado por uma função de Hann, também chamada de janela de Hann. Essa função desempenha um papel importante na redução de efeitos de descontinuidade nas bordas dos segmentos, que poderiam introduzir artefatos inde-

sejados no domínio da frequência. Sem a aplicação de uma janela adequada, a segmentação do sinal pode gerar descontinuidades abruptas nas bordas, resultando no vazamento espectral (spectral leakage), um fenômeno que interfere na precisão da Transformada de Fourier ao distribuir energia do sinal para frequências vizinhas.

Como ilustrado na Figura 17, essa operação consiste na multiplicação de cada amostra do segmento pelo valor correspondente da função de janelamento, resultando em uma transição suavizada nas extremidades do sinal. Essa suavização melhora a precisão da análise espectral, pois evita que variações bruscas no domínio do tempo impactem negativamente a decomposição do sinal no domínio da frequência.

Figura 17 – Aplicação da Função Hann em um Segmento



Fonte: Elaborada pelo autor.

Até este ponto, o processamento do sinal de áudio resultou em segmentos discretos com extremidades suavizadas, minimizando descontinuidades que poderiam comprometer a análise. A implementação dessas operações é detalhada no Algoritmo 1, que divide o sinal de entrada S em janelas sucessivas de tamanho fixo W, aplicando-se a função de janela f_{window} em cada uma para suavizar as bordas. O algoritmo percorre o sinal, extraindo os segmentos conforme o deslocamento definido pelo passo $stepSize$, que depende da taxa de sobreposição α. Os segmentos resultantes são armazenados na estrutura $Segments$.

Algoritmo 1 Segmentação e Janelamento do Sinal de Áudio

Entrada: Sinal de entrada S, tamanho da janela W, sobreposição α, função de janela f_{window}
Saída: Lista de segmentos janelados $Segments$

$stepSize \leftarrow W \times (1 - \alpha)$
$numWindows \leftarrow \frac{|S|-W}{stepSize} + 1$
$Segments \leftarrow \emptyset$
for $winIdx \leftarrow 0$ **to** $numWindows - 1$ **do**
    $start \leftarrow winIdx \times stepSize$
    $segment \leftarrow S[start : start + W]$
    for $i \leftarrow 0$ **to** $W - 1$ **do**
        $segment[i] \leftarrow segment[i] \times f_{window}(i, W)$
    end for
    Append $segment$ to $Segments$
end for
return $Segments$

Fonte: Elaborado pelo autor.

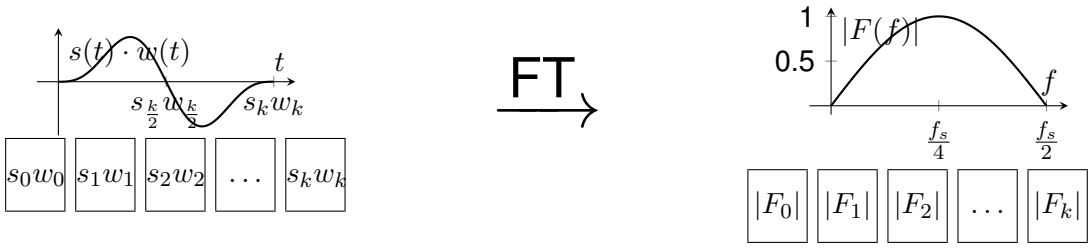
5.2 Espectrograma

Na sequência do processo de janelamento e da aplicação da função de janela Hann, o próximo passo foi a transformação de cada segmento do sinal de áudio para o domínio da frequência. Para isso, foi aplicada a Transformada de Fourier (FT) a cada segmento individualmente, convertendo as informações do domínio do tempo para o domínio da frequência. Esse processo, quando realizado de forma sequencial em segmentos sucessivos ao longo do tempo, é conhecido como Transformada de Fourier de Tempo Curto (STFT - Short-Time Fourier Transform).

O resultado da STFT é um espectrograma, uma matriz tempo-frequência que representa a evolução das componentes espectrais do sinal ao longo do tempo. No espectrograma, cada linha representa uma janela temporal do sinal de áudio, e cada coluna corresponde a um bin de frequência, indicando a intensidade das componentes espectrais em diferentes momentos. Assim, a distribuição da energia do sinal ao longo do tempo e das frequências pode ser visualizada de maneira estruturada, permitindo identificar padrões e características sonoras.

A Figura 18, ilustra esse processo, mostrando como um segmento janelado do sinal foi submetido à FT, resultando em um espectro de frequência. A magnitude espectral obtida para cada segmento é armazenada como uma linha do espectrograma, com suas colunas correspondendo aos bins de frequência. Esse processo é repetido para todos os segmentos do sinal, preenchendo progressivamente a matriz e permitindo a análise visual da evolução das frequências ao longo do tempo.

Figura 18 – Construção da Primeira Linha do Espectrograma a partir da FT



$\|F_{00}\|$	$\|F_{01}\|$	$\|F_{02}\|$	$\cdots$	$\|F_{0k}\|$
$\|F_{10}\|$	$\|F_{11}\|$	$\|F_{12}\|$	$\cdots$	$\|F_{1k}\|$
$\|F_{20}\|$	$\|F_{21}\|$	$\|F_{22}\|$	$\cdots$	$\|F_{2k}\|$
$\vdots$	$\vdots$	$\vdots$	$\ddots;$	$\vdots$
$\|F_{n0}\|$	$\|F_{n1}\|$	$\|F_{n2}\|$	$\cdots$	$\|F_{nk}\|$

Fonte: Elaborada pelo autor.

A Transformada de Fourier aplicada a cada segmento do sinal retorna um número complexo para cada bin de frequência, contendo uma parte real e uma parte imaginária. Para a construção do espectrograma, utilizamos apenas o módulo desses coeficientes complexos, que representa a intensidade da energia em cada frequência, eliminando a necessidade de interpretar separadamente as partes real e imaginária. O módulo é obtido pelo cálculo da raiz quadrada da soma dos quadrados dessas componentes, conforme descrito na Equação 5.1:

$$|F(f)| = \sqrt{\mathsf{Re}(F(f))^2 + \mathsf{Im}(F(f))^2} \tag{5.1}$$

Nesta etapa, além de obter o espectrograma, também identificamos os valores máximo e mínimo das magnitudes espectrais ao longo do tempo. Essas informações serão utilizadas na normalização dos dados nas próximas etapas, garantindo que os valores sejam escalonados em um intervalo padronizado, permitindo descartar frequências com intensidades abaixo de um determinado limiar. O Algoritmo 2 formaliza esse processo, recebendo os segmentos janelados do sinal e retornando o espectrograma juntamente com os valores extremos da magnitude espectral.

Algoritmo 2 Cálculo do Espectrograma a partir dos Segmentos Janelados

Entrada: Lista de segmentos janelados $Segments$, tamanho da janela W
Saída: Espectrograma $Spec$, magnitude máxima M_{max} e magnitude mínima M_{min}

$numWindows \leftarrow |Segments|$
$numBins \leftarrow \frac{W}{2} + 1$
$Spec \leftarrow matrix(numWindows, numBins)$
$M_{max} \leftarrow 0$
$M_{min} \leftarrow \infty$
for $winIdx \leftarrow 0$ **to** $numWindows - 1$ **do**
    $F \leftarrow \mathsf{FT}(Segments[winIdx])$
    for $fIdx \leftarrow 0$ **to** $numBins - 1$ **do**
        $Magnitude \leftarrow \sqrt{\mathsf{Re}(F[fIdx])^2 + \mathsf{Im}(F[fIdx])^2}$
        $Spec[winIdx, fIdx] \leftarrow Magnitude$
        $M_{max} \leftarrow \max(M_{max}, Magnitude)$
        $M_{min} \leftarrow \min(M_{min}, Magnitude)$
    end for
end for
return $Spec, M_{max}, M_{min}$

Fonte: Elaborado pelo autor.

5.3 Detecção de Picos

Após a construção do espectrograma, a próxima etapa do processamento consistiu na detecção de picos. Esses picos representam frequências de maior intensidade ao longo do tempo e são essenciais para a extração de características do sinal. A identificação dessas regiões de alta energia permite reduzir a complexidade da representação espectral, focando apenas nos pontos mais relevantes para o reconhecimento sonoro.

A detecção de picos é realizada percorrendo a matriz do espectrograma e analisando a intensidade relativa de cada bin de frequência em uma região local. Um ponto do espectrograma é considerado um pico se sua magnitude for maior que um limiar definido e superior aos valores vizinhos dentro de um raio pré-determinado.

O método assegura que apenas os pontos de maior energia sejam selecionados, eliminando ruídos e componentes espectrais menos significativos. A Figura 19 ilustra esse processo, destacando uma região do espectrograma onde um pico é avaliado em relação aos seus vizinhos, demonstrando visualmente a lógica empregada na detecção.

Figura 19 – Detecção de Pico no Espectrograma

$\|F_{00}\|$	$\|F_{01}\|$	$\|F_{02}\|$	$\|F_{03}\|$	$\|F_{04}\|$	$\|F_{05}\|$	$\|F_{06}\|$
$\|F_{10}\|$	$\|F_{11}\|$	$\|F_{12}\|$	$\|F_{13}\|$	$\|F_{14}\|$	$\|F_{15}\|$	$\|F_{16}\|$
$\|F_{20}\|$	$\|F_{21}\|$	$\|F_{22}\|$	$\|F_{23}\|$	$\|F_{24}\|$	$\|F_{25}\|$	$\|F_{26}\|$
$\|F_{30}\|$	$\|F_{31}\|$	$\|F_{32}\|$	$\|\mathbf{F_{33}}\|$	$\|F_{34}\|$	$\|F_{35}\|$	$\|F_{36}\|$
$\|F_{40}\|$	$\|F_{41}\|$	$\|F_{42}\|$	$\|F_{43}\|$	$\|F_{44}\|$	$\|F_{45}\|$	$\|F_{46}\|$
$\|F_{50}\|$	$\|F_{51}\|$	$\|F_{52}\|$	$\|F_{53}\|$	$\|F_{54}\|$	$\|F_{55}\|$	$\|F_{56}\|$
$\|F_{60}\|$	$\|F_{61}\|$	$\|F_{62}\|$	$\|F_{63}\|$	$\|F_{64}\|$	$\|F_{65}\|$	$\|F_{66}\|$

Fonte: Elaborada pelo autor.

O Algoritmo 3 formaliza esse processo, percorrendo o espectrograma para identificar os picos. Ele utiliza um limiar ajustável, baseado nas magnitudes máxima e mínima do espectrograma. Além disso, um raio de detecção é aplicado para comparar cada bin de frequência com seus vizinhos, assegurando que apenas os pontos de maior intensidade dentro de sua região sejam considerados picos. Os picos detectados são armazenados como uma lista de coordenadas temporais e espectrais, acompanhadas de sua respectiva magnitude.

Algoritmo 3 Detecção de Picos Locais no Espectrograma

Entrada: Espectrograma $Spec$, magnitude máxima M_{max}, magnitude mínima M_{min}, limiar de detecção $threshold$, raio de detecção r

Saída: Lista de picos detectados $Peaks$

$numRows \leftarrow |Spec|$
$numCols \leftarrow |Spec[0]|$
$Peaks \leftarrow \emptyset$
$scaledThreshold \leftarrow M_{min} + (M_{max} - M_{min}) \times \frac{threshold}{100}$
for $i \leftarrow r$ **to** $numRows - r - 1$ **do**
    for $j \leftarrow r$ **to** $numCols - r - 1$ **do**
        $currentValue \leftarrow Spec[i, j]$
        if $currentValue < scaledThreshold$ **then**
            continue
        end if
        $isPeak \leftarrow$ **true**
        for $di \leftarrow -r$ **to** r **do**
            for $dj \leftarrow -r$ **to** r **do**
                if $di = 0$ **and** $dj = 0$ **then**
                    continue
                end if
                if $currentValue \leq Spec[i + di, j + dj]$ **then**
                    $isPeak \leftarrow$ **false**
                    break
                end if
            end for
            if not $isPeak$ **then**
                break
            end if
        end for
        if $isPeak$ **then**
            Append $(i, j, currentValue)$ to $Peaks$
        end if
    end for
end for
return $Peaks$

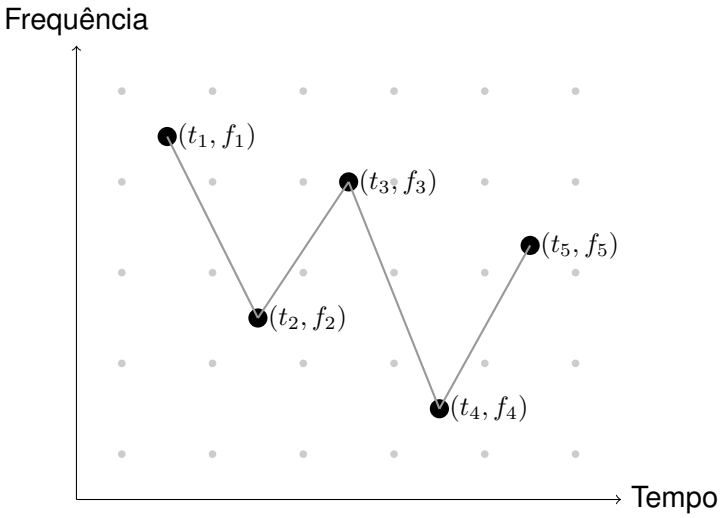
Fonte: Elaborado pelo autor.

5.4 Criação de Hashes

Com a detecção de picos no espectrograma, a última etapa do algoritmo consiste na geração de hashes, que servem como identificadores únicos para cada trecho do sinal de áudio. Esses hashes são importantes para a indexação e recuperação de informações do sinal, permitindo a identificação de trechos sonoros.

A criação dos hashes é baseada nos picos detectados no espectrograma, estabelecendo associações entre diferentes frequências dominantes ao longo do tempo. Esses pares de frequências formam assinaturas únicas do sinal. A Figura 20 ilustra esse processo, mostrando como os picos são conectados.

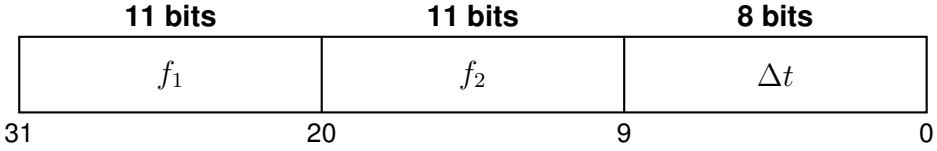
Figura 20 – Associação de Picos no Espectrograma



Fonte: Elaborada pelo autor.

O processo de combinação de picos é realizado conectando picos próximos no espectrograma, desde que respeitem um limite máximo de tempo entre eles. A partir dessas conexões, geramos um hash de 32 bits para cada combinação. A Figura 21 ilustra a estrutura desse hash, que é construído utilizando 11 bits para a frequência do primeiro pico (f_1), 11 bits para a frequência do segundo pico (f_2) e 8 bits para o intervalo temporal (Δt) entre os dois picos.

Figura 21 – Estrutura do Hash de 32 bits



Fonte: Elaborada pelo autor.

Como os bins de frequências no espectrograma variam até 2048, 11 bits são suficientes para representar cada frequência, pois permitem armazenar até $2^{11} = 2048$ valores distintos. Já o intervalo de tempo entre os picos é armazenado com 8 bits, possibilitando representar até 256 valores distintos.

O Algoritmo 4 executa esse processo, utilizando os picos previamente extraídos para gerar os hashes. Cada hash armazena informações sobre a posição temporal e espectral dos picos, permitindo que trechos do sinal sejam comparados e identificados posteriormente. Para a geração dos hashes, foi considerado um delta máximo de tempo de 200 janelas, um valor compatível com a estrutura de 32 bits do hash, no qual 8 bits foram alocados para representar essa diferença temporal.

Algoritmo 4 Geração de Hashes a partir dos Picos Detectados

Entrada: Lista de picos $Peaks$, delta máximo Δt_{max}
Saída: Lista de hashes $Hashes$

sort($Peaks$)
$Hashes \leftarrow \emptyset$
$numPeaks \leftarrow |Peaks|$
for $i \leftarrow 0$ **até** $numPeaks - 1$ **do**
    $(t_1, f_1) \leftarrow Peaks[i]$
    for $j \leftarrow i + 1$ **até** $numPeaks - 1$ **do**
        $(t_2, f_2) \leftarrow Peaks[j]$
        $\Delta t \leftarrow |t_2 - t_1|$
        if $\Delta t > \Delta t_{max}$ **then**
            break
        end if
        $f_1 \leftarrow f_1 \wedge 0x7FF$
        $f_2 \leftarrow f_2 \wedge 0x7FF$
        $\Delta t \leftarrow \Delta t \wedge 0xFF$
        $hash \leftarrow (f_1 \ll 21) \vee (f_2 \ll 10) \vee \Delta t$
        Adicionar $(hash, t_1)$ a $Hashes$
    end for
end for
return $Hashes$

Fonte: Elaborado pelo autor.

5.5 Aplicação do Algoritmo e Armazenamento de Hashes

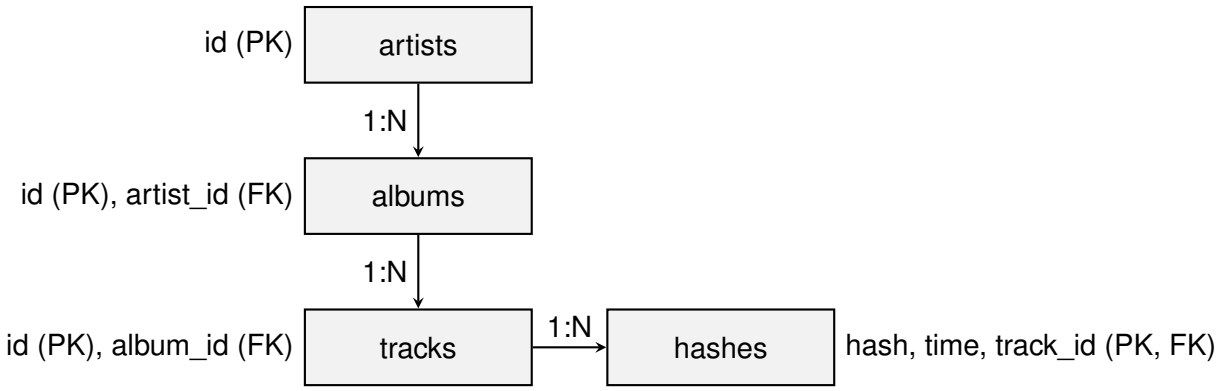
Para avaliar a precisão do algoritmo de geração de hashes, este foi aplicado a um conjunto de faixas musicais extraídas da API do serviço iTunes, abrangendo trechos de 30 segundos provenientes de álbuns de 36 artistas representativos de distintos gêneros musicais. O experimento teve como finalidade a construção de um banco de dados de impressões digitais sonoras, possibilitando a posterior identificação e correspondência de segmentos de áudio previamente armazenados.

A implementação do Algoritmo de Impressão Digital foi realizada em C++, seguindo as etapas descritas nas seções anteriores. Para o cálculo da Transformada de Fourier, foi utilizada a biblioteca FFTW3, enquanto a leitura dos arquivos de áudio foi conduzida com a biblioteca JUCE. O processamento das faixas considerou um único canal e uma taxa de amostragem de 44.100 Hz. Cada música foi submetida às etapas de segmentação, geração de espectrograma, extração de picos e cálculo dos hashes.

Para facilitar a entrada de dados no sistema, foi desenvolvida uma API utilizando o framework Crow em C++. A API oferece duas rotas principais. A primeira rota recebe uma música e seus metadados, persiste os metadados no banco de dados e gera os hashes correspondentes, que também são armazenados. A segunda rota recebe um trecho de áudio, gera os hashes desse trecho e busca no banco de dados as músicas que possuem maior correspondência com os hashes gerados.

Os hashes gerados foram armazenados em um banco de dados relacional PostgreSQL, juntamente com os metadados das músicas e dos artistas. A estrutura do banco de dados foi projetada para permitir a rápida recuperação das impressões digitais, por meio da criação de índices. A modelagem relacional adotada é composta por quatro tabelas principais, conforme ilustrado na Figura 22.

Figura 22 – Modelagem Relacional para Armazenamento dos Hashes



Fonte: Elaborada pelo autor.

Os metadados armazenados no banco de dados incluíram informações sobre artistas, álbuns, faixas e as impressões digitais extraídas de cada música. A tabela artists contém identificadores e nomes dos artistas, enquanto a tabela albums armazena informações sobre os álbuns, incluindo o gênero primário, a data de lançamento, o país de origem, a arte da capa e os direitos autorais.

As faixas individuais foram catalogadas na tabela tracks, que mantém um vínculo com o álbum correspondente e inclui um link para prévia da música. Por fim, a tabela hashes contém as impressões digitais sonoras geradas pelo algoritmo, associando cada hash ao respectivo instante temporal dentro da faixa analisada.

Após a catalogação da base de músicas, foram realizados testes para avaliar a precisão do algoritmo na identificação de faixas a partir de trechos curtos. Para isso, foram selecionadas 1000 músicas aleatórias da base de dados, e a partir de cada uma delas foram extraídos trechos entre 1 e 5 segundos.

Cada trecho foi submetido à API, que gerou os hashes correspondentes e os consultou no banco de dados, buscando recuperar o ID da música com o maior número de correspondências. Foi considerado um acerto quando a música retornada pelo sistema correspondia ao identificador da faixa original do trecho submetido.

Após a realização dos testes, foram efetuadas consultas no banco de dados para analisar a relação entre os gêneros musicais armazenados e as taxas de acerto e erro do algoritmo. A distribuição dos acertos e erros foi comparada com a quantidade de faixas de cada gênero, buscando identificar padrões que pudessem indicar maior ou menor eficácia na identificação de determinadas categorias musicais.

6 Resultados

Este capítulo apresenta os resultados obtidos a partir da execução do algoritmo, incluindo estatísticas da base de dados e a avaliação do desempenho na identificação de músicas. Primeiramente, são discutidos os dados armazenados no banco de dados, abrangendo a quantidade de músicas catalogadas, a distribuição dos gêneros musicais e a quantidade de hashes gerados para cada categoria. Em seguida, são analisados os resultados da taxa de acerto do algoritmo, considerando diferentes durações de trechos de áudio.

6.1 Análise da Base de Dados

A base de dados construída armazenou um volume expressivo de informações, permitindo uma análise dos registros processados. No total, foram catalogadas 18.601 faixas musicais, distribuídas em 1.280 álbuns de 36 artistas de diferentes gêneros musicais. Além disso, foram geradas aproximadamente 116 milhões de hashes, correspondendo às impressões digitais das músicas processadas. A Tabela 2 apresenta um resumo quantitativo desses dados, destacando a distribuição dos registros e o armazenamento ocupado por cada conjunto.

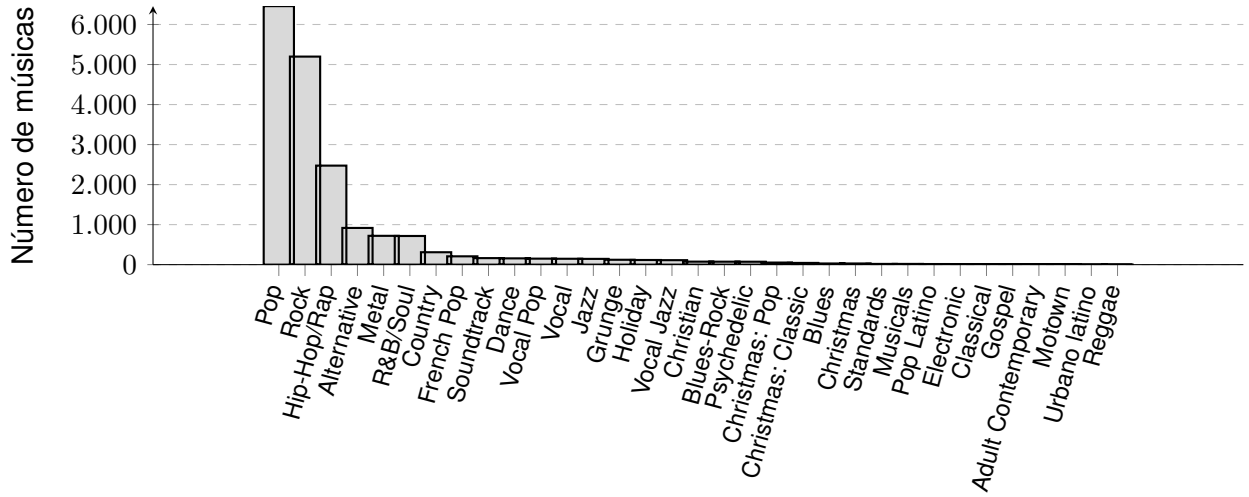
Tabela 2 – Quantidade de Registros e Armazenamento no Banco de Dados

Categoria	**Quantidade de Registros**	**Armazenamento**
Faixas (*tracks*)	18.601	5.296 kB
Hashes (*hashes*)	116.625.319	11 GB
Artistas (*artists*)	36	32 kB
Álbuns (*albums*)	1.280	1.496 kB

Fonte: Elaborada pelo autor.

Entre os gêneros musicais catalogados, observou-se uma predominância de Rock, Pop e Hip-Hop/Rap, que juntos constituíram a maioria da base de dados. A diversidade de estilos presentes na base permitiu a aplicação do algoritmo em diferentes contextos musicais, abrangendo variações na estrutura espectral e rítmica das faixas. A Figura 26 apresenta a distribuição dos gêneros na base, demonstrando a variação no número de faixas por categoria.

Figura 23 – Distribuição dos Gêneros Musicais na Base de Dados

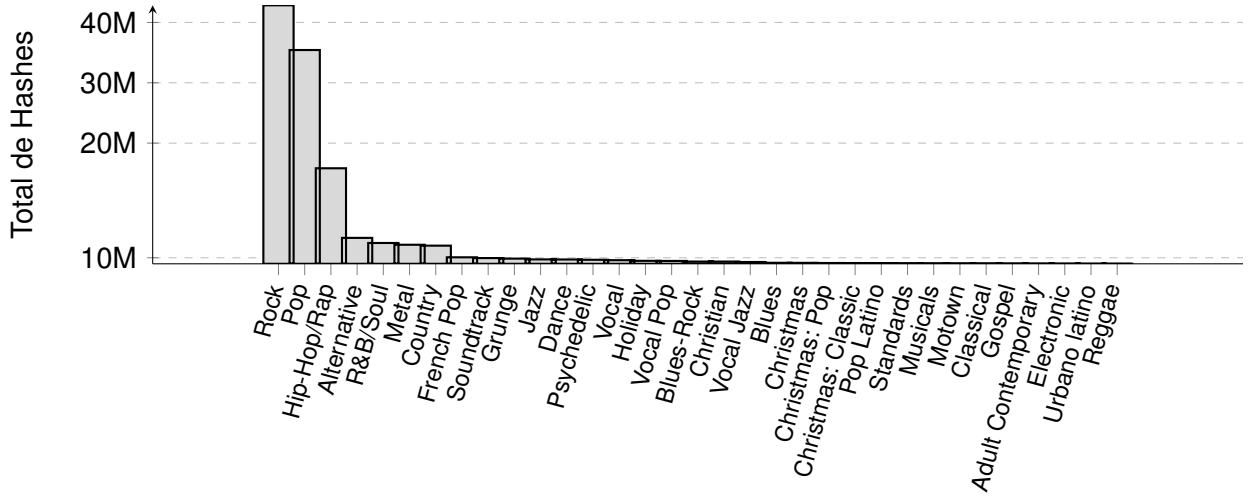


Fonte: Elaborada pelo autor.

A quantidade de hashes gerados por gênero musical reflete a disposição dos picos de frequência ao longo do espectrograma das faixas processadas. Como esperado, gêneros com mais faixas, como Rock e Pop, apresentaram um volume mais elevado de hashes. No entanto, a Figura 24 indica que, apesar de o Pop possuir mais músicas catalogadas, a quantidade de hashes gerados foi inferior à do Rock.

Esse comportamento sugere que a distribuição espectral das músicas influenciou diretamente a quantidade de picos detectados no espectrograma e, consequentemente, a quantidade de hashes gerados. O número de impressões digitais armazenadas variou entre os gêneros, refletindo características estruturais distintas, como densidade espectral e variação nas transições de frequência ao longo do tempo.

Figura 24 – Quantidade de Hashes Gerados por Gênero Musical

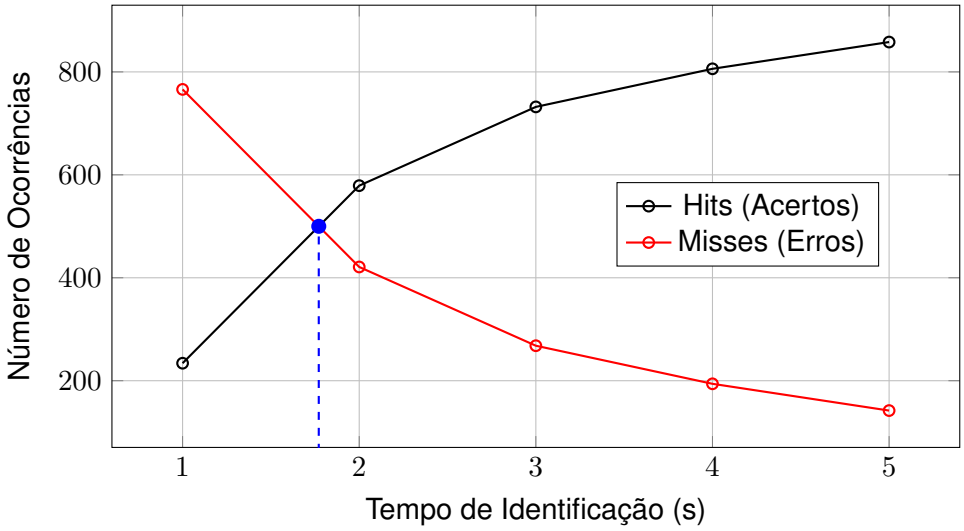


Fonte: Elaborada pelo autor.

6.2 Análise da Precisão do Algoritmo

A avaliação da precisão do algoritmo foi realizada considerando diferentes tempos de identificação, variando de 1 a 5 segundos. O objetivo foi analisar o impacto do tempo disponível na taxa de acertos e erros durante o processo de identificação. A Figura 25 apresenta a relação entre o tempo de identificação e o número de acertos e erros registrados nos testes realizados com 1.000 músicas selecionadas aleatoriamente.

Figura 25 – Relação entre tempo de identificação e precisão.



Fonte: Elaborada pelo autor.

Observa-se ainda um ponto de interseção entre as curvas de acertos e erros, destacado na Figura 25, indicando o momento em que o número de ocorrências se igualou para ambos os casos. Esse ponto foi identificado em 1,8 segundos, sugerindo que, para trechos de áudio com duração superior a esse valor, o algoritmo apresentou uma melhora na precisão, resultando em um número maior de acertos em relação aos erros.

A Tabela 3 apresenta a relação entre o tempo de identificação e a taxa de acertos e erros. A porcentagem de acertos foi maior para trechos de maior duração, ou seja, o número de erros foi reduzido à medida que o tempo disponível para análise aumentou.

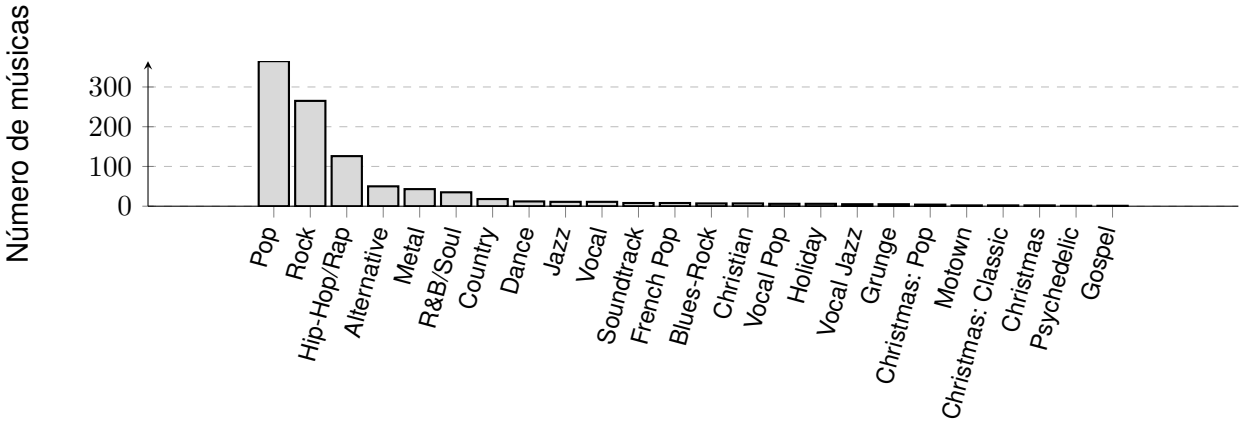
Tabela 3 – Porcentagem de acertos e erros por tempo de identificação

Tempo de Identificação (s)	**Acertos (%)**	**Erros (%)**
1	23,4%	76,6%
2	57,9%	42,1%
3	73,2%	26,8%
4	80,6%	19,4%
5	85,8%	14,2%

Fonte: Elaborada pelo autor.

As faixas selecionadas para o teste apresentaram uma distribuição variada de gêneros musicais, conforme a Figura 26. Gêneros mais populares na base de dados, como Pop e Rock, compõem a maior parcela das 1.000 faixas selecionadas.

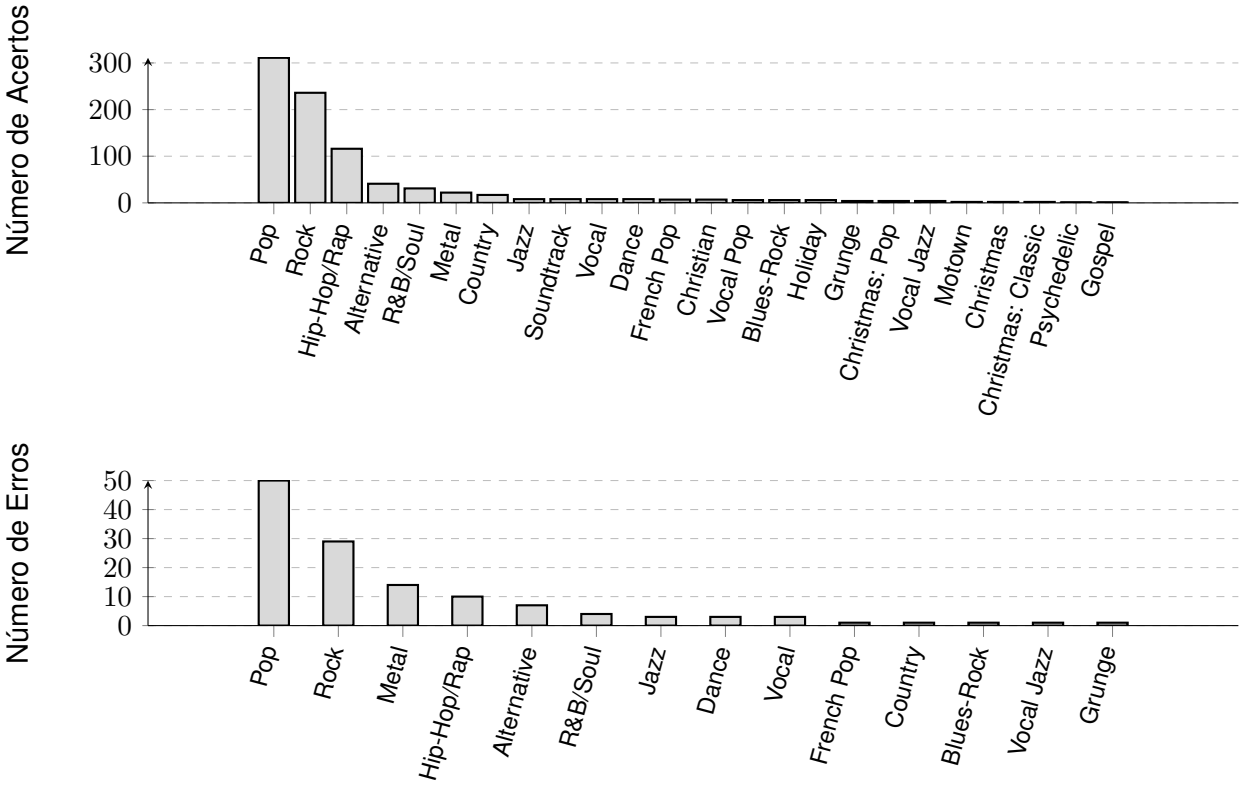
Figura 26 – Distribuição dos Gêneros Musicais nas Faixas de Teste



Fonte: Elaborada pelo autor.

A Figura 27 exibe a distribuição de acertos e erros do algoritmo por gênero musical. Nota-se que os gêneros com maior número de faixas entre as 1.000 selecionadas, como Pop e Rock, também apresentaram os maiores números de acertos.

Figura 27 – Distribuição de Acertos e Erros por Gênero Musical



Fonte: Elaborada pelo autor.

7 Considerações Finais

Este trabalho teve como objetivo desenvolver e avaliar um algoritmo de identificação de áudio baseado em impressões digitais sonoras, permitindo a recuperação de trechos musicais a partir de uma base de dados. Para isso, foi implementado um algoritmo que envolveu a segmentação do sinal, o cálculo do espectrograma, a detecção de picos e a geração de hashes, possibilitando a criação de uma representação compacta de cada faixa musical.

A implementação foi realizada em C++. A base de dados foi construída a partir de 18.601 faixas musicais, organizadas em 1.280 álbuns de 36 artistas de diferentes gêneros musicais. Os hashes gerados foram armazenados em um banco de dados PostgreSQL, estruturado para eficiente recuperação dos trechos identificados.

Os testes foram conduzidos sobre 1.000 músicas selecionadas aleatoriamente, avaliando a precisão do algoritmo para trechos de 1 a 5 segundos. Os resultados mostraram que a taxa de acertos aumentou com a duração do trecho, e a interseção das curvas de acertos e erros indicou que, a partir de 1,8 segundos, o algoritmo passou a apresentar mais acertos do que erros. Além disso, a análise dos dados evidenciou que a quantidade de hashes gerados variou entre os gêneros musicais.

O estudo demonstrou a eficácia do método proposto na identificação de músicas a partir de trechos curtos, evidenciando sua viabilidade para detectar potenciais violações de direitos autorais. Além disso, este trabalho também serve como uma síntese dos principais algoritmos de processamento de áudio, apresentando um panorama das técnicas utilizadas para extração, representação e reconhecimento de impressões digitais sonoras, possibilitando sua aplicação em diferentes contextos de monitoramento e proteção de conteúdo.

Ressalta-se que este trabalho foi apresentado na Mostra Específica de Trabalhos e Aplicações (META) do CEFET-MG, Campus Timóteo, onde foi exposto um projeto que utiliza GPU para acelerar o cálculo do espectrograma. No entanto, essa abordagem não foi empregada na presente pesquisa, pois a entrada de áudio utilizada é pequena e seu processamento já ocorre de maneira eficiente na CPU, sem a necessidade de paralelização via GPU.

7.1 Trabalhos Futuros

Este trabalho apresentou uma solução para a extração e armazenamento de impressões digitais. No entanto, ainda há diversas oportunidades para aprimoramento e expansão do método. Entre as direções futuras, destacam-se:

- **Avaliação de Diferentes Configurações de Parâmetros**: Análise do impacto de diferentes valores para os principais parâmetros do sistema, incluindo:
  - `windowSize`: tamanho da janela utilizada na análise espectral.
  - `overlap`: taxa de sobreposição entre janelas sucessivas.

  - `threshold`: limite utilizado para a detecção de picos espectrais.
  - `detectionRadius`: tamanho da região considerada ao identificar picos.
  - `maxHashDeltaTime`: intervalo máximo permitido entre hashes correlacionados.

- **Aplicação de Técnicas para Remoção de Ruído**: Implementação de filtros para reduzir interferências e melhorar a precisão na extração de impressões digitais, tornando o sistema mais robusto contra variações nos áudios analisados.
- **Exploração de Paralelismo**: Investigação de técnicas de paralelização para acelerar a identificação de picos, utilizando arquiteturas multi-thread ou processamento em GPU.
- **Análise da Consideração de Vales Além de Picos**: Exploração da viabilidade de incluir vales espectrais, além dos picos, no processo de extração de características. Isso pode fornecer informações complementares que melhorem a discriminação entre diferentes trechos de áudio.

Referências

ARLOTTA, L. D. R. B. *Direitos Autorais na Era Digital: A Transformação do Mercado Fonográfico e os Desafios Jurídicos Provenientes da Tentativa de Regulamentação*. 2017. Monografia — Universidade Federal Fluminense, Niterói, RJ, 2017. Citado na página 11.

BORGES, A. N.; RODRIGUES, C. G. *Introdução à Física Acústica*. 1a edição. ed. São Paulo, SP: Editora Livraria da Física, 2017. 150 p. ISBN 9788578614843. Citado nas páginas 13, 14 e 15.

CANO, P. et al. A review of audio fingerprinting. *Journal of VLSI Signal Processing*, Springer Science + Business Media, Inc., v. 41, 2005. Citado nas páginas 11 e 18.

DEBNATH, L.; BHATTA, D. *Integral Transforms and Their Applications*. 2nd. ed. Boca Raton, FL: Chapman & Hall/CRC, Taylor & Francis Group, 2007. Library of Congress Cataloging-in-Publication Data QA432.D36 2006 515'.723–dc22. ISBN 1-58488-575-0. Citado na página 19.

DECHICCHIS, J. E. *A Music Identification Algorithm which Utilizes the Fourier Analysis of Audio Signals*. 2016. Candidate Number: 006832-0004. Citado nas páginas 23 e 24.

ELMASRI, R.; NAVATHE, S. *Sistemas de Banco de Dados*. 4ª. ed. São Paulo, Brasil: Pearson, 2005. Inclui referências bibliográficas e índice. ISBN 978-8588639171. Citado nas páginas 21 e 22.

GOOGLE. *How Content ID works*. 2024. Accessed: May 2024. Disponível em: <https://support.google.com/youtube/answer/2797370>. Citado na página 11.

HIGGINBOTHAM, J. *Designing Great Web APIs: Creating Business Value Through Developer Experience*. First edition. Sebastopol, CA, USA: O'Reilly Media, Inc., 2015. ISBN 978-1-491-92459-4. Citado na página 22.

JESUS, W. de. *Identificação e Classificação de Acordes Musicais aplicando a Transformada de Fourier*. 2019. Trabalho de Conclusão de Curso (Mestrado Profissional em Matemática) — Universidade Federal de São João del-Rei (UFSJ), São João del-Rei, MG, 2019. Sociedade Brasileira de Matemática (SBM). Citado nas páginas 25 e 26.

LYONS, R. G. *Understanding Digital Signal Processing*. 3rd. ed. Upper Saddle River, NJ: Prentice Hall, 2011. Includes bibliographical references and index. ISBN 0-13-702741-9. Citado na página 20.

OPPENHEIM, A. V.; WILLSKY, A. S.; NAWAB, S. H. *Signals and Systems*. 2nd. ed. Upper Saddle River, NJ: Prentice-Hall, 1996. (Prentice-Hall Signal Processing Series). ISBN 0-13-814757-4. Citado nas páginas 19 e 20.

OSGOOD, B. *Lecture Notes for EE 261: The Fourier Transform and its Applications*. 2007. Stanford University, Electrical Engineering Department. Disponível em: <https://see.stanford.edu/materials/lsoftaee261/book-fall-07.pdf>. Citado na página 19.

PROAKIS, J. G.; MANOLAKIS, D. G. *Digital Signal Processing: Principles, Algorithms, and Applications*. 3rd. ed. Upper Saddle River, New Jersey: Prentice-Hall, Inc., 1996. ISBN 978-0133942897. Citado nas páginas 16, 17 e 21.

WANG, A. An industrial strength audio search algorithm. In: *ISMIR 2003, 4th International Conference on Music Information Retrieval, Baltimore, Maryland, USA, October 27-30, 2003, Proceedings*. [S.l.: s.n.], 2003. Citado nas páginas 24 e 25.

WAZLAWICK, R. S. *Metodologia de pesquisa para ciência da computação*. 6. reimpressão. ed. Rio de Janeiro: Elsevier, 2009. ISBN 978-85-352-3522-7. Citado na página 27.