**UNIVERSIDADE FEDERAL DE PERNAMBUCO**

**CENTRO DE INFORMÁTICA**

**CURSO DE BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO**

**Livisghton Kleber da Silva Araújo**

**Análise do Uso de *Chromagrams* na Classificação Automática de Acordes Musicais**

**RECIFE**

**2019**

**UNIVERSIDADE FEDERAL DE PERNAMBUCO**

**CENTRO DE INFORMÁTICA**

**CURSO DE BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO**

**Livisghton Kleber da Silva Araújo**

**Análise do Uso de *Chromagrams* na Classificação Automática de Acordes Musicais**

Monografia apresentada ao Centro de Informática (CIN) da Universidade Federal de Pernambuco (UFPE), como requisito parcial para conclusão do Curso de Engenharia da Computação, orientada pelo professor Carlos Alexandre Barros de Mello.

**RECIFE**

**2019**

**UNIVERSIDADE FEDERAL DE PERNAMBUCO**

**CENTRO DE INFORMÁTICA**

**CURSO DE BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO**

**Livisghton Kleber da Silva Araújo**

**Análise do Uso de *Chromagrams* na Classificação Automática de Acordes Musicais**

Monografia submetida ao corpo docente da Universidade Federal de Pernambuco, defendida e aprovada em 02 de dezembro de 2019.

Banca Examinadora:

Orientador

Carlos Alexandre Barros de Mello

Doutor

Examinador

Tsang Ing Ren

Doutor

Dedico esse trabalho a minha mãe Jacilene Rodolfo da Silva, ao meu pai Lauribel Cavalcanti de Araujo.

**AGRADECIMENTOS**

Gostaria de agradecer primeiramente a Deus, a minha família que sem dúvida alguma é meu maior apoio, sem eles eu nada seria. Em especial, a minha mãe Jacilene Rodolfo da Silva e meu pai Lauribel Cavalcante de Araujo e minha noiva Taisa Ramos da Silva por todo apoio durante este percurso.

Gostaria de agradecer também aos meus amigos por me ajudar a chegar até aqui e em especial ao meu primo José Cleyton da Silva e meu amigo Alex Alves do Nascimento Silva, que me ajudaram do inicio ao fim no desenvolvimento deste projeto.

Queria agradecer ao meu orientador Carlos Alexandre de Barros de Mello pelo apoio e suporte, além de, seus conselhos que foram fundamentais para a construção deste trabalho. Por fim, agradeço também a Universidade Federal de Pernambuco e o Centro de Informática, por todo conhecimento adquirido.

“A persistência é o caminho do êxito.”

Charles Chaplin

**RESUMO**

A cifra é uma notação musical utilizada para indicar qual acorde será executado por um instrumento em conjunto a melodia de uma música. Músicos menos experientes geralmente recorrem a sites ou repositórios de cifras, devido não ter a habilidade de reconhecer acordes pelos seus próprios ouvidos para executar alguma canção. No entanto, existe uma grande quantidade de músicas que não estão catalogadas nessas plataformas dificultando assim o aprendizado de determinadas músicas. Para auxiliar no aprendizado desses músicos, foi construído um modelo computacional utilizando redes neurais para reconhecer acordes. Basicamente, o modelo é composto por duas etapas: 1. Extração de características utilizando *chromagrams*, 2. Estimativas de acordes por meio de uma MLP. Na fase de extração de características foram testados quatro tipos de *chromagrams,* CP *Features*, CLP *Features*, CENS *Features* e CRP *Features*. Os resultados experimentais extraídos de mais de 150 músicas dos Beatles e Queen indicaram que a aplicação desses diferentes tipos de *chromas* como extração de características obtém-se quase o mesmo resultado, ou seja, a variação da acurácia para os quatro tipos de *chromagrams* foi muito baixa. Mas a acurácia da MLP para reconhecer acordes ficou entre 63,1% e 64,4% que é um bom resultado.

**Palavras-chave**: Reconhecimento de Acordes, Redes Neurais, *Chromagram*.

**ABSTRACT**

The cipher is a musical notation used to indicate which chord will be played by an instrument together with the melody of a song. Less experienced musicians often turn to cipher sites or repositories because they lack the ability to recognize chords in their own ears to perform a song. However, there are a lot of songs that are not cataloged on these platforms making it difficult to learn certain songs. To assist in the learning of these musicians, a computational model was built using neural networks to recognize chords. Basically, the model consists of two steps: 1. Feature extraction using chromagrams, 2. Chord estimations using an MLP. In the feature extraction phase four types of chromagrams were tested, CP Features, CLP Features, CENS Features and CRP Features. Experimental results from over 150 Beatles and Queen songs indicated that applying these different types of chromas as feature extraction yields almost the same result, ie the accuracy variation for the four types of chromagrams was very low. Experimental results from over 150 Beatles and Queen songs indicated that applying these different types of chromas as feature extraction yields almost the same result, ie the accuracy variation for the four types of chromagrams was very low. But the accuracy of MLP to recognize chords was between 63.1% and 64.4% which is a good result.

**Keywords**: Chord Recognition, Neural networks, *Chromagram* .

Sumário

[1. Introdução 15](#_Toc24929216)

[1.1. Objetivos 16](#_Toc24929217)

[1.2. Objetivos Específicos 16](#_Toc24929218)

[1.3. Estrutura do Projeto 16](#_Toc24929219)

[2. Fundamentação teórica 18](#_Toc24929220)

[2.1. Conceitos Musicais 18](#_Toc24929221)

[2.2. Conceitos de Processamento de Sinais 20](#_Toc24929222)

[2.3. Conceitos de](#_Toc24929223) *[Chromagram](#_Toc24929223)* [21](#_Toc24929223)

[2.4. Conceitos de Redes Neurais 23](#_Toc24929224)

[3. Metodologia do Estudo 26](#_Toc24929225)

*[3.1.](#_Toc24929229)* [Tipos de](#_Toc24929229) *[Chromagrams](#_Toc24929229)* [26](#_Toc24929229)

[3.2. Rede MLP 29](#_Toc24929230)

[4 Experimento e Análise 30](#_Toc24929231)

[4.1 Construção da base de dados 30](#_Toc24929232)

[4.2 Experimentos 32](#_Toc24929233)

[4.3 Resultados 33](#_Toc24929235)

[5 Conclusão 35](#_Toc24929236)

[5.1 Contribuições e Dificuldades 35](#_Toc24929237)

[5.2 Trabalhos Futuros 35](#_Toc24929238)

[Referências 36](#_Toc24929239)

**Lista de Figuras**

[Figura 1: Escala cromática crescente, ilustrando as notas (DÓ, DÓ#, RÉ, RÉ#, MI, FÁ, FÁ#, SOL, SOL#, LÁ, LÁ#, SÍ), nesta ordem. Fonte: próprio autor. 19](#_Toc24922596)

[Figura 2: Escala cromática decrescente, ilustrando as notas (SÍ, SIb, LÁ, LAb, SOL, SOLb, FÁ, FÁb, MI, MIb, RÉ, RÉb, DÓ), nesta ordem. Fonte: próprio autor. 19](#_Toc24922597)

[Figura 3: Representação de acordes em partituras 20](#_Toc24922598)

[Figura 4: Representação de acordes na música](#_Toc24922599) *[Let It Be](#_Toc24922599)* [dos Beatles. Imagem capturada do site cifra Club. 20](#_Toc24922599)

[Figura 5: Variações de acordes de tétrades. A letra T na última coluna representa a tónica do acorde, ou seja, a nota de 1º grau. Fonte: http://aguitarra.com.br 21](#_Toc24922600)

[Figura 6: Passos para construção de um](#_Toc24922601) *[chromagram](#_Toc24922601)* [23](#_Toc24922601)

[Figura 7: Representação gráfica de uma estrutura do perceptron. 25](#_Toc24922602)

[Figura 8: Arquitetura de uma Rede Neural MLP. Fonte: encurtador.com.br/lmpY1 26](#_Toc24922603)

[Figura 9: Representação do CP](#_Toc24922604) *[Feature.](#_Toc24922604)* [27](#_Toc24922604)

[Figura 10: Representação do CLP](#_Toc24922605) *[Feature](#_Toc24922605)*[. 28](#_Toc24922605)

[Figura 11: Representação do CENS](#_Toc24922606) *[Feature](#_Toc24922606)*[. 29](#_Toc24922606)

[Figura 12: Representação do CRP](#_Toc24922607) *[Feature](#_Toc24922607)*[. 30](#_Toc24922607)

[Figura 14: Fluxo para construção do banco de dados. 32](#_Toc24922608)

[Figura 15: Exemplo das informações contidas nos arquivos .lab. Este exemplo mostra as informações dos 10 primeiros segundos da música](#_Toc24922609) *[Let it Be](#_Toc24922609)* [da banda](#_Toc24922609) *[The Beatles](#_Toc24922609)*[. 32](#_Toc24922609)

**LISTA DE TABELAS**

[Tabela 1: Número de ocorrência de acordes dentro dos arquivos .lab. 32](#_Toc24901771)

[Tabela 2: Resultado do classificador MLP para cada](#_Toc24901772) *[chormagram](#_Toc24901772)*[. 34](#_Toc24901772)

**LISTA DE SIGLAS**

|  |  |  |
| --- | --- | --- |
| **Sigla** |  | **Significado** |
| C |  | Acorde de Dó |
| D |  | Acorde de Ré |
| E |  | Acorde de Mi |
| F |  | Acorde de Fá |
| G |  | Acorde de Sol |
| A |  | Acorde de Lá |
| B |  | Acorde de Sí |
| N |  | Silêncio |
| M ou maj |  | Acorde maior |
| m |  | Acorde menor |
| dim |  | Acorde diminuto |
| # |  | Sustenido |
| b |  | Bemol |
| Hz |  | Hertz |

# Introdução

A cifra é uma notação musical utilizada para indicar os acordes que serão executados por um instrumento (como piano, violão ou ukulele, por exemplo). Ela é bastante utilizada acima das letras, tablaturas ou partituras, indicando o acorde a ser tocado em conjunto a melodia [1]. É bastante comum que músicos amadores ou não profissionais não possuam a habilidade de distinguir os acordes pelos seus próprios ouvidos, fazendo-os recorrerem a sites de cifras como: cifra Club[[1]](#footnote-1), cifras[[2]](#footnote-2) ou pega cifra[[3]](#footnote-3) por exemplo. No entanto, existem várias músicas que não estão cadastradas na base de dados dessas ferramentas, dificultando assim, o aprendizado dessas canções por parte dos músicos menos experientes.

Segundo VELOSO e FEITOSA [2], “...a capacidade de identificar, nomear ou produzir a frequência de um estímulo tonal sem o auxílio de um tom de referência”, caracterizam músicos que possuem “ouvido absoluto”. Similarmente, SANTOS JUNIOR [3] fala sobre “ouvido relativo” em músicos que consiste em; “...não se limita a identificar as notas, mas também as características intervalares que permitem entender a música e as relações entre as notas, acordes e melodias”. Em virtude disto, esses tipos de músicos não apresentam problemas ou dificuldades em identificar notas musicais e reproduzi-las. Por outro lado, a ocorrência de ter um ouvido absoluto é rara, de acordo com VELOSO e FEITOSA [2]. Já para possuir um ouvido relativo requer treino e isso leva tempo para aperfeiçoar.

Com base nisto, este trabalho tem como objetivo construir um modelo computacional que seja capaz de estimar acordes musicais dado um arquivo de música. Para isso, foram estudados modelos baseados em redes neurais artificiais e técnicas de processamento de áudio como o *chroma feature* para estimar os acordes em músicas. Com isso, espera-se colaborar no auxílio do aprendizado de músicos iniciantes e amadores que não apresentam nenhuma das características mencionadas anteriormente. Além disso, fornecer conhecimento na área de extração de características de acordes musicais.

## Objetivos

O objetivo deste trabalho, é criar um modelo computacional que seja capaz de estimar acordes músicas por meio de arquivos de músicas digitais. Além disso, utilizar este modelo para avaliar sua eficiência sobre algumas técnicas de extrações de características de acordes dentro de uma composição musical. Com isso, espera-se colaborar no auxílio do aprendizado de músicos iniciantes e amadores que não tem a habilidade de identificar as acordes músicas.

## Objetivos Específicos

* Aplicar a técnica de *chroma feature* sobre os arquivos de áudio.
* Treinar a máquina utilizando o algoritmo de aprendizagem: *Multilayer Perceptrons* (MLP).
* Avaliar o desempenho dos tipos de *chromagrams* sobre a MLP.
* Analisar os resultados por meio de tabelas.

## Estrutura do Projeto

Os demais capítulos deste trabalho possuem a seguinte estrutura:

**No capítulo 2**, é apresentado a fundamentação teórica necessária para o projeto. Nesse capitulo, é abordado conceitos básicos sobre música, processamento de sinais, *chromagram* e por fim redes neurais.

**No capítulo 3,** é apresentado a metodologia de estudo do projeto. Nessa parte, é exibido os diferentes tipos de *chormagrams* e a configuração utilizada na rede neural MLP.

**No capítulo 4,** é aborado a análise e os resultado obtidos nos experimentos. Aqui, é mostrado em detalhes como foi construído a base de dados para os experimentos, as configurações necessárias para executar os experimentos e os resultados obtidos.

**No capítulo 5,** fecha o projeto fazendo uma conclusão dos resultados obtidos nos experimentos. Além disso, é comentado quais rumos o trabalho poderá seguir no futuro.

# Fundamentação teórica

Neste capítulo, são introduzidos alguns conceitos básicos necessários para o entendimento deste trabalho. Inicialmente, são explicados conceitos básicos de notas e acordes musicais. Em seguida, são apresentadas teorias computacionais de processamento de sinais e redes neurais.

## Conceitos Musicais

O elemento mínimo sonoro dentro de uma composição musical é chamado de nota musical. Este elemento apresenta um único modo de vibração do ar e está relacionado a uma frequência (medida em Hz), onde descreve em termos físicos se ela é mais grave ou mais aguda [4].

O nosso sistema musical, ocidental, é composto por doze notas musicais diferentes, sendo sete as notas naturais (Dó, Ré, Mi, Fá, Sol, Lá e Si) e cinco as acidentadas que, dependendo da escala, podem ser (Dó#, Ré#, Fá#, Sol# e Lá#), representando as notas sustenidas ou (Réb, Mib, Solb, Láb e Sib), representando as notas bemóis. Estas doze notas podem ser vistas graficamente nas Figuras 1 e 2.



Figura 1: Escala cromática crescente, ilustrando as notas (DÓ, DÓ#, RÉ, RÉ#, MI, FÁ, FÁ#, SOL, SOL#, LÁ, LÁ#, SÍ), nesta ordem. Fonte: próprio autor.

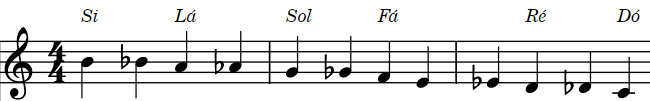


Figura 2: Escala cromática decrescente, ilustrando as notas (SÍ, SIb, LÁ, LAb, SOL, SOLb, FÁ, FÁb, MI, MIb, RÉ, RÉb, DÓ), nesta ordem. Fonte: próprio autor.

Já um acorde pode ser formado por meio de uma combinação de, pelo menos, três ou mais notas tocadas simultaneamente. A representação gráfica de um acorde em uma partitura ou cifra, pode ser vista nas Figuras 3 e 4, respectivamente.

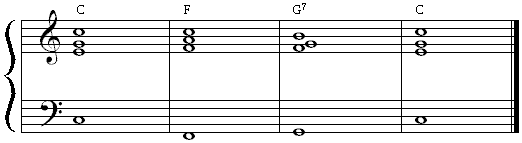


Figura 3: Representação de acordes em partituras

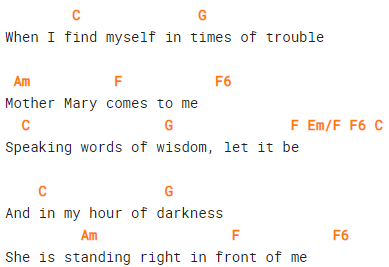
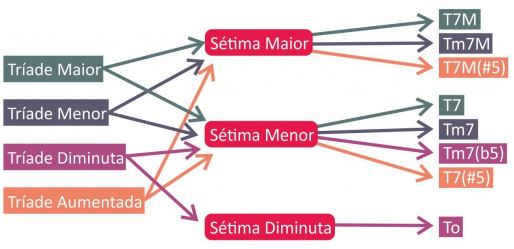


Figura 4: Representação de acordes na música *Let It Be* dos Beatles. Imagem capturada do site cifra Club.

Os acordes formados por três notas são chamados de tríades, onde, na maioria das vezes, são constituídos pelas notas de 1º, 3º e 5º grau. Por exemplo: um acorde de Dó maior é formado pelas notas Dó, Mi e Sol que representa o 1º, 3º e 5º graus respectivamente. A variação das notas de 3º e 5º graus geram novos acordes, onde os mais conhecidos são: acordes maiores, menores, com quartas (SUS4), quinta aumentada e diminuta [5], por exemplo: C, Cm, Csus4, C(#5), C(b5), respectivamente.

Já os acordes formados por quatro notas são chamados de tétrade, onde aparecem as notas de 7º grau. Exemplo: um acorde de Dó com sétima é formado pelas notas Dó, Mi, Sol e Sib, na qual representa 1º, 3º, 5º e 7º graus respectivamente. Similarmente aos acordes formados por tríades, a variação da nota do 7º grau também gera novos acordes, como podemos ver na Figura 5.



**Figura 5:** Variações de acordes de tétrades. A letra T na última coluna representa a tónica do acorde, ou seja, a nota de 1º grau.Fonte: <http://aguitarra.com.br>

## Conceitos de Processamento de Sinais

O processamento de sinais está ligado à natureza do sinal e a aplicação, onde normalmente consiste na análise e/ou modificação de sinais, utilizando teoria fundamental, aplicações e algoritmos de forma a obter informações ou apropriando o mesmo para uma aplicação específica. Existem duas formas para fazer o processamento de sinais, uma é a analógica e a outra é a digital [6]. Neste trabalho, é utilizado o sinal no formato digital.

Para poder trabalhar com um sinal digital, partindo de um sinal analógico, é preciso discretizá-lo. Para isso, o sinal analógico passará por duas fases que são amostragem e quantização (geralmente os CDs – *Compact Discs* - utilizam 16 bit/amostra) [7].

Na fase de amostragem, é calculada a quantidade de amostras em um dado período de tempo, ou seja, haverá uma frequência (taxa de amostragem ou taxa de Nyquist) associada a um período de tempo , que resulta um conjunto finito de amostras para um intervalo de tempo. A expressão matemática que representa esta fase é dada por:

(Eq. 1)

Segundo o teorema de amostragem de Nyquist mostrado na Equação 2, a taxa de amostragem deve ser pelo menos duas vezes a maior frequência . O ouvido humano consegue diferenciar sons entre a faixa de frequência de 20 Hz e 22.050 Hz. Portanto, a máxima frequência do ouvido humano é de 22.050 Hz; então a frequência de Nyquist deve ser 44.100 Hz. Dessa forma, são captadas 44.100 amostras de áudio a cada segundo.

(Eq. 2)

A transformada de Fourier é uma das ferramentas mais fundamentais deste processo, pois ela consegue a representar um sinal que está no domínio do tempo no domínio da frequência. Com foi mostrado na Seção 2.1, cada nota está associada a uma frequência, fazendo com que a nota seja grave para frequências baixas e agudas para frequências altas. A representação da transformada discreta de Fourier (*Discrete Fourier Transform -* DFT) [6] para N pontos é dada por:

|  |
| --- |
| (Eq.3) |

onde: e .

Já a transformada discreta inversa de Fourier é dada por:

|  |
| --- |
| (Eq.4) |

onde: .

## Conceitos de *Chromagram*

O *chroma* (representado em *chromagrams*) é uma ferramenta poderosa de extração de características no contexto de reconhecimento de acordes musicais [7] e foi introduzida por Fujishima [9]. Ele pode ser utilizado para extrair características de harmonia [9] ou melodia [10] de uma composição musical. Geralmente, o *chroma* é representado por um vetor de 12 posições, onde cada posição do vetor pode ser interpretada como uma nota musical, por exemplo: {Dó, Do#, Ré, Re#, Mi, Fá, Fá#, Sol, Sol#, Lá, Lá#, Sí}. Neste trabalho, o *chroma* foi utilizado para obter informações apenas da harmonia da música.

Existem várias formas de representar um acorde em um vetor de *chroma.* Por exemplo:

* Atribuir valor 1 quando a nota pertence ao acorde e 0 caso contrário. Exemplo, o acorde de C é formado pelas notas dó, mi e sol, então o vetor de *chroma*, usando esta abordagem, seria {1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0}.
* Atribuir uma probabilidade para cada nota, onde 1 seria a probabilidade máxima da nota pertencer ao acorde e 0 caso contrário. Exemplo, usando como base o acorde de C, novamente, um possível vetor gerado seria {1, 0.01, 0.01, 0.01, 0.9, 0, 0, 0.9, 0.01, 0.01, 0.01, 0.01}.

As principais etapas de construção de um *chromagram* podem ser vistas na Figura 6.

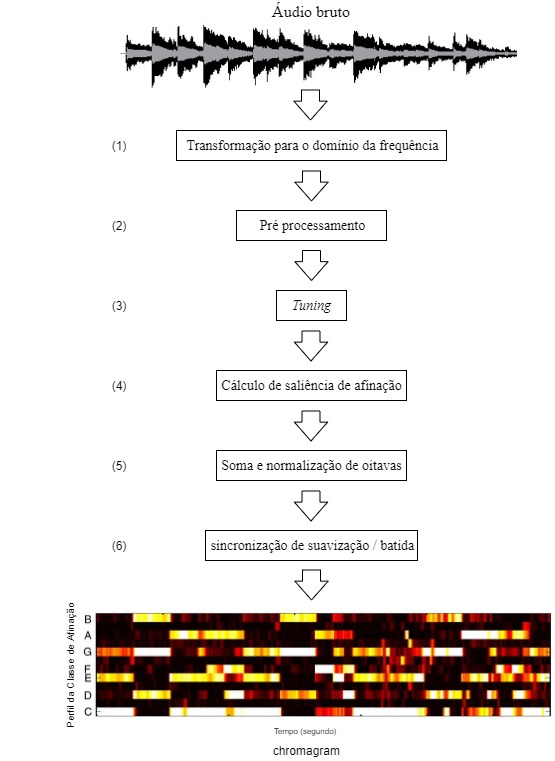


Figura 6: Passos para construção de um *chromagram*

Partindo de um sinal bruto discretizado no domínio do tempo, o primeiro passo para construir um *chromagram* é converter o sinal para domínio da frequência e calcular sua magnitude. Existem algumas técnicas para representar o sinal no domínio da frequência de forma simples como DFT. No entanto, a DFT traz descrição global das frequências presentes no áudio e, de acordo com [7], os pesquisadores que estudam a estimação automática de acordes (*Automatic Chord Estimation* - ACE) estão interessados nas variações harmônicas locais e, por isso, geralmente é utilizada a Transformada de Fourier de tempo curto (*Short-Time Fourier Transform* - STFT) que calcula as magnitudes de frequência em uma janela deslizante através do sinal.

O segundo passo, pré-processamento, tenta eliminar parte inútil do espectro como o espectro de fundo[11][12] e alguns harmônicos que podem facilmente confundir técnicas de extração de características [13][14].

De acordo com Sheh e Ellis, existem algumas faixas de músicas que não estão afinadas no tom padrão A4 = 440 Hz [15]. Então, o terceiro passo, *tuning*, aplica algoritmos como [16][18] para ajustar a afinação dessas músicas.

O quarto passo, cálculo de saliência de afinação, captura a saliência da classe de afinação fazendo o mapeamento do espectro obtido nos passos anteriores com a saliência de uma nota.

No estágio final do cálculo do *chromagram*, soma e normalização de oitavas, é feita a soma de todas as saliências pertencentes a uma mesma classe. Essa soma é normalizada para produzir uma matriz de recurso do *chromagram* que captura a evolução da afinação do áudio ao longo do tempo. Por fim, suavização /sincronização de batida, é uma etapa de pós processamento para minimizar as frequentes mudanças de acordes e ruído.

## Conceitos de Redes Neurais

Para compreender como funciona o modelo *Multilayer Perceptrons* (MLP), primeiramente é necessário entender o funcionamento de um *perceptron*. Semelhante ao cérebro humano que consiste em uma rede de neurônios, uma rede neural consiste em uma rede de neurônios artificiais, chamados de *perceptrons*.

O *perceptron* foi criado por Frank Rosenblatt por volta dos anos 1957. A estrutura do perceptron pode ser vista na Figura 8, onde um conjunto de entradas combinadas aos seus respectivos pesos são classificadas em 0 ou 1, ou -1 e 1, dependendo da função de ativação que for usada. A Equação 5 mostra a representação matemática de um perceptron levando em consideração o seu *bias*, já a Equação 6 mostra a função de ativação de um neurônio [18].

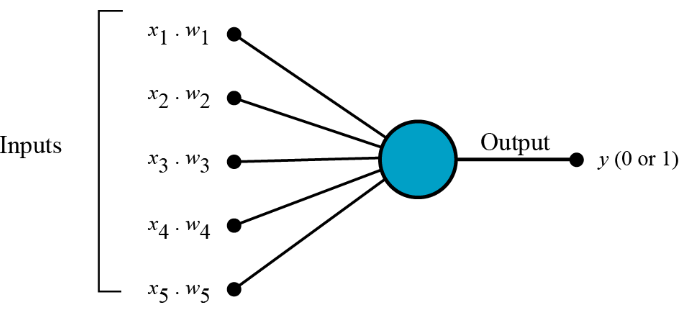


Figura 7: Representação gráfica de uma estrutura do perceptron.

(Eq.5)

(Eq.6)

O grande problema desse modelo, é que ele se limita em resolver problemas onde o conjunto de dados seja linearmente separável. Para resolver problemas mais complexos e semelhante ao cérebro humano que é composto por vários neurônios, existem modelos que são compostos por vários perceptrons, sendo um deles chamado de chamado de Perceptron multicamadas (*Multi Layer Perceptron* - MLP).

A MLP consiste em uma estrutura em camadas (uma camada de entrada, uma ou mais camadas escondidas/intermediárias e uma camada de saída), cada uma pode conter um ou mais perceptron. Além disso, o sinal de entrada é propagado para frente conforme os neurônios de cada camada. Por isso, as MLPs são consideradas redes neurais do tipo *feedforward* [21].

O treinamento de uma MLP é feito de forma supervisionada utilizando o algoritmo gradiente descendente (*backpropagation*). Este algoritmo é dividido em duas etapas, a primeira é a fase de ida, onde um padrão é apresentado a camada de entrada e, a partir desta camada as unidades calculam sua resposta e produz um conjunto de saída. A segunda fase é a de volta, onde partir da saída gerada na fase anterior é calculado um erro, este erro é propagado a partir da camada de saída até a camada de entrada, e os pesos das conexões das camadas internas vão sendo justados [21].

Por fim, a Figura 9 é um exemplo de arquitetura de uma MLP, neste exemplo na camada de entrada apresenta dois neurônios, existem duas camadas intermediárias com quatro neurônios e na camada de dois neurônios.

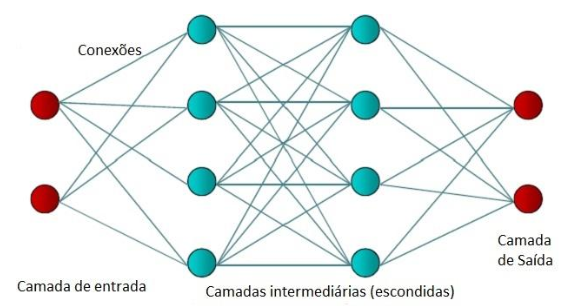


Figura 8: Arquitetura de uma Rede Neural MLP. Fonte: [encurtador.com.br/lmpY1](https://www.researchgate.net/publication/265552060_O_USO_DE_REDES_NEURAIS_PARA_A_ANALISE_E_CONCESSAO_DE_CREDITO_FLAVIO_IZO_-flavioflavioizocom_INSTITUTO_FEDERAL_DO_ESPIRITO_SANTO_-IFES)

# Metodologia do Estudo

Nesta seção, são apresentadas as configurações utilizadas no classificador MLP e os diferentes tipos de *chromagrams* utilizados no experimento deste projeto.

# Tipos de *Chromagrams*

Esse projeto faz uma comparação entre o desempenho de quatro tipos de *chromagrams* utilizando um classificador MLP. Os *chromagrams* utilizados nesse estudo foram: *Chroma-Pitch**features*(ou CP *Features*), *Chroma-Log-Pitch**Features* (ou CLP *Features*), *Chroma Energy Normalized Statistics Features* (CENS *Features*) e *Chroma DCT-Reduced log Pitch* (CRP *Feature*)*.* Além disso, é tomado como referência para mostrar a variação dos *chromagrams*, os 10 primeiros segundos da música *Let it Be* da banda *The Beatles*. Este trecho da música utilizado contém apenas o piano, ou seja, não há outro instrumento e nem há presença do cantor. Foi tomado como base [7][20][21] e [22] para explicar os passos seguintes, onde é possível encontrar mais detalhes sobre os mesmos.

* CP *Features:*

A extração de características baseado em *chroma,* tem como objetivo representar a energia em um tempo curto do sinal em cada uma das 12 classes de afinação. Normalmente, estas características são obtidas por meio de uma STFT. Em seguida, é feito um mapeamento dessas características com o vetor que contém as classes de afinação, o *chroma*. Por fim, geralmente é aplicado uma normalização nos vetores resultantes com o objetivo de minimizar o quase silêncio ou algum ruído muito pequeno. Estes passos podem ser vistos de forma mais geral na Figura 6 na Seção 2.3 e a Figura 9 fornece a representação do CP *Features*.

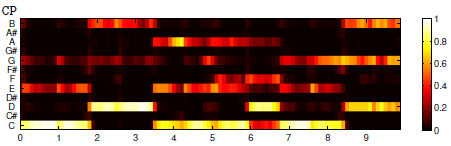


Figura 9: Representação do CP *Features.*

* CLP *Features:*

O CLP tem como objetivo ajustar a faixa do sinal original para melhorar a clareza de transitórios fracos, especialmente nas regiões de alta frequência. Para isso, aplica-se frequentemente uma compressão logarítmica ao calcular as características do áudio.

Para construir um CLP, primeiramente é calculado as características de afinação do arquivo de áudio. Em seguida, a representação da afinação é logaritmizada substituindo cada entrada pelo , onde é uma constante positiva que especifica a extensão da compactação logarítmica. Depois, com a representação logarítmica projetar os vetores de afinação resultantes em um vetor cromatográfico. Como exemplo ilustrativo, a Figura 10 fornece a representação do CLP *Features*.

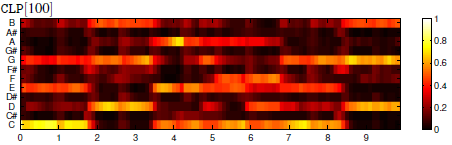


Figura 10: Representação do CLP *Features*.

* CENS *Features:*

O CENS *Features* é um aperfeiçoamento do CP *Features* que trata algumas propriedades musicais, como dinâmica, timbre, articulação, execução de um grupo de notas e micro desvios temporais. Para ser robusto a essas variações, foi adicionado um grau de abstração aos recursos do *Chroma-Pitch*, considerando estatísticas de tempo curto sobre a distribuição de energia nas faixas de *chroma*.

Segue os passos para construção do CENS:

* + - 1. Normalização: Primeiramente, são normalizadas as características do *chroma* para capturar diferentes tipos de intensidade ou dinâmica do som. Quando a distribuição de energia é muito baixa ou silenciosa, o vetor do *chroma* é substituído por um vetor uniformemente distribuído, se a norma não exceder determinado limite.
      2. Quantização: Os componentes do vetor *chroma* normalizado são quantizados com base em limiares algoritmicamente escolhidos para simular a intensidade de volume do som no ouvido humano. Devido a isso, é necessário introduzir algum tipo de compactação logarítmica semelhante ao CLP.
      3. Suavização: Os vetores quantizados agora são convoluídos com uma janela *Hanning* de tamanho fixo, onde. Essa etapa tem como objetivo reduzir o erro local.
      4. *Downsampling:* Reduz as amostras dos vetores de características resultantes por um valor especifico , com o objetivo de aumentar a eficiência computacional para o próximo módulo.
      5. Normalização: Por fim, os vetores de características são normalizados por .

Como exemplo ilustrativo, a Figura 11 fornece a representação do CENS *Features*.

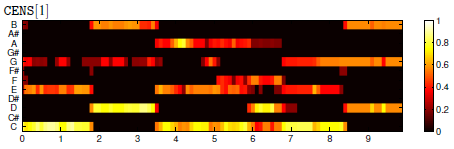


Figura 11: Representação do CENS *Features*.

* CRP *Features:*

A ideia geral deste *chromagram* é descartar as informações relacionadas ao timbre. Para construir um CRP*,* é aplicada uma compressão logarítmica e, em seguida, é utilizada a Transfomada Discreta do Cosseno (*Discrete Cosine Transform* - DCT) para representar a afinação em uma escala logarítmica. Em seguida, apenas os coeficientes superiores são mantidos e é aplicada uma DCT inversa sobre esses coeficientes resultantes. Por fim, projetam-se os vetores de afinação resultantes em um vetor cromatográfico de 12 dimensões. Os coeficientes superiores a serem mantidos são especificados por um parâmetro .

Semelhante ao CENS, o CRP tem uma etapa de suavização, onde convolve um vetor de recursos com vetores de recursos vizinhos. O comprimento da janela da convolução é definido por . Além disso, também apresenta uma etapa de redução de amostragem onde a variável é responsável por isso.

A Figura 12 ilustra o CRP *Features*.

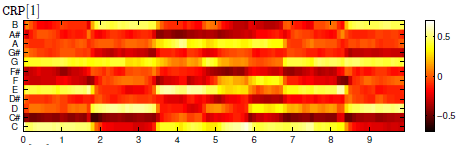


Figura 12: Representação do CRP *Feature*.

# Rede MLP

A escolha do classificador para este projeto foi com o objetivo de simplesmente avaliar as técnicas de pré-processamento que estão sendo utilizadas no reconhecimento de acordes musicais. Então, foi escolhida uma rede neural chamada de MLP, para ser treinada e comparar sua classificação para os diversos tipos de *chromagrams*.

A configuração da MLP utilizada nos experimentos foi baseada em trabalhos de detecção de acordes para piano [23][24]. Onde a rede tem uma camada escondida com 18 neurônios. Além disso, a rede foi treinada com 1000 iterações por 30 vezes utilizando o *backpropagation*.

# Experimento e Análise

Esta seção mostra a metodologia utilizada para construir a base de dados, como foram feitos os experimentos para comparar os diferentes tipos de *chromagrams* sobre um classificador MLP e por fim, é mostrado os resultados obtidos na análise.

## Construção da base de dados

A base de dados utilizado neste projeto contempla 200 músicas no total, onde 20 são músicas da banda *Queen* e 180 da banda *The Beatles*; esta base está disponível no site *isophonics* [25].

Antes de iniciar a construção da base de dados para alimentar a MLP, foi percebido que haviam notas iguais porem com o nome diferente na base de dados da *isophonics,* isso é devido ao campo harmônico de um tom musical[26]. Por exemplo, uma música no tom de F# e uma música no tom de Db podem aparecer em seus campos harmônicos os acordes de C# e Db; no entanto, eles representam o mesmo acorde, análogo as notas músicas, como pode ser visto nas Figura 1 e Figura 2 na seção 2.1. Para resolver este problema, foi deixado de lado a tonalidade musical, pois não é relevante para este contexto, e todos os acordes bemóis foram convertidos para sustenidos.

A construção da base de dados para alimentar a MLP foi feita com base nos passos mostrado na Figura 14. Partido das músicas, onde , foram gerados *chromagrams*, ou seja, um chromagram para cada arquivo de música. Em seguida, esses *chromagrams,* foram mapeados com os arquivos .*lab* para poder representar vetores de *chromas* com duração de 0,1 segundo; isso aumentou significativamente a quantidade de acordes para ser usado no treinamento da rede. Cada música na base de dados dispõem de um arquivo .lab, esses arquivos contêm informações como o início e fim dos acordes ao longo da música, como pode ser visto na Figura 15. Por fim, foram selecionados os vinte acordes que apresentaram as maiores ocorrências em todas as músicas, os acordes com maior ocorrência são mostrados na Tabela 1. Foi necessário fazer esta escolha para não prejudicar o classificador, devido ao desbalanceamento dos dados. No banco de dados A, mostrado na Figura14, foi o banco utilizado na MLP, já normalizado contendo apenas os vinte acordes escolhidos, ou seja, nele há 3500 exemplos de cada um dos vinte acordes totalizando 70000 exemplos. No banco de dados B, não foi feito nenhum tratamento, apenas foi salvo com a intensão de utiliza-lo em trabalhos futuros.

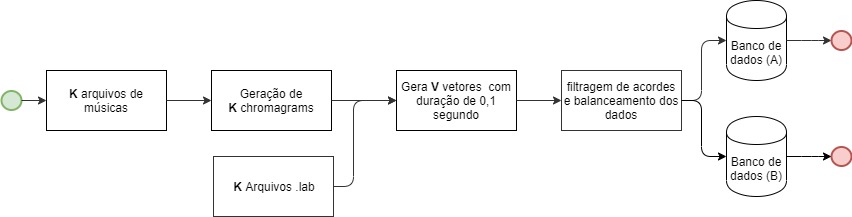


Figura 14: Fluxo para construção do banco de dados.

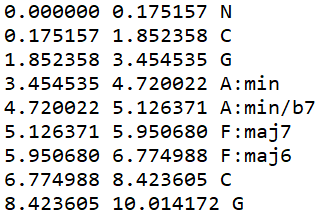


Figura 15: Exemplo das informações contidas nos arquivos .lab. Este exemplo mostra as informações dos 10 primeiros segundos da música *Let it Be* da banda *The Beatles*.

Tabela 1: Número de ocorrência de acordes dentro dos arquivos .lab.

|  |  |
| --- | --- |
| **Acordes** | **Número de ocorrência dentro de todos os arquivos .lab** |
| A | 1723 |
| G | 1482 |
| D | 1441 |
| C | 1047 |
| E | 1028 |
| B | 526 |
| F | 510 |
| N | 487 |
| A:min | 435 |
| E:min | 407 |
| Bb | 357 |
| B:min | 329 |
| F#:min | 293 |
| D:min | 219 |
| Eb | 192 |
| D/5 | 191 |
| F# | 190 |
| Ab | 175 |
| C#:min | 169 |
| G:7 | 164 |

## Experimentos

Os experimentos foram realizados utilizando as linguagens MATLAB e Python. Foi utilizado o MATLAB para geração dos *chromagrams*, porque já há implementações dos mesmos disponível em [28][29]. As implementações desses *chromas* foram realizadas por Müller e Ewert no trabalho [23]. Já para a implementação da MLP, foi utilizada a biblioteca *sklearn* de Python. Estas implementações também estão públicas no site do *Github* [30].

Primeiramente, foi utilizado o programa *Format Factory* na configuração padrão para converter todos os arquivos no formato Mp3 para Wav. Em seguida, foi gerado as *Features* de cada arquivo Wav por meio da função *converteWavtoPitch.m*, este processo foi feito apenas uma vez, porque todos os *chromas* partem deste ponto.

Para geração do *CP Features*, foi utizada a função *pitch\_to\_chroma.m*, com a omissão da compressão logarítma, gerando assim duas bases de dados chamadas de bd\_CP.cvs e bd\_CP1.csv.

Para gerar o CLP *Features,* foi utilizada a mesma função do CP, mas agora aplicando a compressão logarítma , gerando também duas bases de dados, bd\_CLP.csv e bd\_CLP1.csv.

Já para gerar o CENS *Features*, foi utilizada a função *pitch\_to\_CENS.m*, onde o tamanho da janela é e redução de amostras , também foi gerado duas bases de dados, bd\_CENS.csv e bd\_CENS.csv1.

Por fim, para gerar o CRP *Features*, foi utilizada a função *pitch\_to\_CRP.m,* onde o tamanho da janela é , redução de amostras , compressão logarítma e os coeficientes superiores . Além disso, foram geradas duas bases de dados, bd\_CRP.csv e bd\_CRP1.csv.

Com a criação da base de dados pronta, pegou-se as bases de dados: bd\_CP, bd\_CLP, bd\_CENS e bd\_CRP para alimentar a MLP. As bases de dados bd\_CP1, bd\_CLP1, bd\_CENS1 e bd\_CRP1, foram geradas apenas com o intuito de utilizar em trabalhos futuros, visto que, não houve tempo para serem analisadas por completo.

Para cada experimento, CP, CLP, CENS e CRP. As bases de dados foram divididas em duas partes com a seguinte divisão 70% para treinamento e 30% para testes. Além disso, a rede foi treinada com o número máximo de iterações 1000 durante 30 vezes.

## Resultados

Avaliaremos o desempenho da MLP sobre os *chromagrams* a partir das instâncias de testes. Como o problema é um típico problema de classificação de multi-classes, então foi utilizada a macro-média (ou *macro avg*), média ponderada (ou *weighted avg*) e a acurácia (ou *accuracy*) para ter uma visão geral sobre as métricas estatísticas que foram aplicadas. As métricas aplicadas para uso de comparação dos resultados foram: *precision*, *recall* e *f1-score.*

Tabela 2: Resultado do classificador MLP para cada *chormagram*.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tipos de *Chromagrams* |  | *Precision* | *Recall* | *f1-score* | *accuracy* |
| CP | ***macro avg*** | 0,645 | 0,642 | 0,641 | 0,641 |
|  | ***weighted avg*** | 0,645 | 0,642 | 0,641 |
| CLP | ***macro avg*** | 0,645 | 0,643 | 0,642 | 0,644 |
|  | ***weighted avg*** | 0,645 | 0,644 | 0,643 |
| CENS | ***macro avg*** | 0,634 | 0,631 | 0,630 | 0,632 |
|  | ***weighted avg*** | 0,633 | 0,632 | 0,631 |
| CRP | ***macro avg*** | 0,632 | 0,630 | 0,630 | 0,631 |
|  | ***weighted avg*** | 0,632 | 0,631 | 0,631 |

Podemos observar na Tabela 2, a rede neural MLP obteve um bom desempenho mesmo sendo um classificador simples, comparado a trabalhos onde se utilizam classificadores mais complexos a diferença da acurácia não é tão grande. No entanto, esse classificador não conseguiu detectar as peculiaridades de cada *chromagram,* ou seja, tanto faz utilizar o algoritmo CP *Features*, CLP *Features*, CENS *Features* ou CRP *Features*, que a taxa de precisão será bem próxima.

# Conclusão

Neste trabalho, estudou-se parte da literatura de extração de características de acordes músicas, com objetivo de avaliar o desempenho de um classificador MLP para diferentes tipos de *chromas*. Para isso, foram construídas bases de dados para cada tipo de chroma

Para gerar os *chromas* foi utilizado implementações e parâmetros já validados.

## Contribuições

Neste trabalho, foi feito uma comparação dos diferentes tipos de *chromagrams* utilizando uma MLP. Os experimentos para geração dos *chromas* foram retirados de trabalhos já validados. Com isso foi possível comparar os desempenhos dos tipos de *chromas* sobre uma MLP. Os resultados deste projeto servirão como base para outros pesquisadores.

## Trabalhos Futuros

Em trabalhos futuros pretendo utilizar outros tipos de classificadores como por exemplo: Rede Neural Convolucional (ou *Convolucional Neural Network* - CNN) para tentar aumentar a precisão no reconhecimento de acordes. Fazer algum tratamento nas bases de dados que foram criadas neste projeto e que não foram utilizadas, com isso aumentaria a quantidade de acorde a serem reconhecidos. Por fim, construir uma aplicação *web* ou *mobile* para geração automática de cifras.

# Referências

1. wikipedia. [Online]: <https://pt.wikipedia.org/wiki/Cifra_(música)>. (Acessado em 01/10/2019).
2. F.Veloso, M.A.Feitosa. "O Ouvido Absoluto: bases neurocognitivas e perspectivas". 2013, p. 357 - 362.
3. P.J. dos Santos Junior. "Ouvido absoluto e ouvido relativo: sua natureza e relevância para a educação musical". In XVI Encontro Regional Sul da ABEM; 2014.
4. Wikipédia. [Online]: <https://pt.wikipedia.org/wiki/Nota_musical>. (Acessado em 13/10/2019).
5. descomplicandoamusica. [Online]: <https://www.descomplicandoamusica.com/triades/>. (Acessado em 13/10/2019).
6. A.Oppenheim, R.W.Schafer. "Discrete-Time Signal Processing": Prentice-Hall; 1989.
7. J. P. d. Santana Neto. (2015). Solução computacional para reconhecimentode harmonias musicais Música - solução computacional para reconheci-mento de harmonias, endereço:http://bdm.unb.br/handle/10483/11336(acesso em 17/11/2019)
8. M.McVicar, R.Santos-Rodriguez, Y.Ni, T.De Bie. "Automatic Chord Estimation from Audio: A Review of the State of the Art". IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2014, p. 556-575.
9. T.Fujishima. "Realtime chord recognition of musical sound: A system using Common Lisp Music", Proc. Int. Comput. Music Conf., 1999, p. 464–467.
10. M.Muller e S.Ewert. "Towards Timbre-Invariant Audio Features for Harmony-Based Music", IEEE Transactions on Audio, Speech, and Language Processing, 2010.
11. H.J.Kallman. "Tone chroma is functional in melody recognition", Perception & Psychophysics, 1979, 26.
12. N.Ono, K.Miyamoto, J.Le Roux, H.Kameoka, e S.Sagayama. “Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram”, 16th European Signal Processing Conference, 2008.
13. J. Reed, Y. Ueda, S. Siniscalchi, Y. Uchiyama, S. Sagayama, and C. Lee, “Minimum classification error training to improve isolated chord recognition,” in Proc. 10th Int. Soc. Music Inf. Retrieval, 2009, pp. 609–614.
14. S.Pauws. “Musical key extraction from audio”, Proc. 5th Int. Soc. Music Inf. Retrieval. 2004: p. 66–69.
15. M.Mauch, e S.Dixon. “Simultaneous Estimation of Chords and Musical Context From Audio”, IEEE Transactions on Audio, Speech, and Language Processing, 2010, p. 1280-1289.
16. A. Sheh and D. Ellis, “Chord segmentation and recognition using em-trained Hidden Markov Models,” in Proc. 4th Int. Soc. Music Inf. Retrieval, 2003, pp. 183–189.
17. C. Harte and M. Sandler, “Automatic chord identification using a quantised chromagram,” in Proc. Audio Eng. Soc., 2005, pp. 291–301.
18. C. Harte, M. Sandler, and M. Gasser, “Detecting harmonic change in musical audio,” in Proc. 1st Workshop Audio Music Comput. Multimedia, 2006, pp. 21–26.
19. L.C.Akshay. towardsdatascience: <https://medium.com/ensina-ai/redes-neurais-perceptron-multicamadas-e-o-algoritmo-backpropagation-eaf89778f5b8>.
20. S.Haykin. "Neural Computation : A Comprehensive Foundation". 2ª Edição, Editora Prentice Hall, 1998.
21. N. Jiang, P. Grosche, V. Konzi e M. Muller, “Analyzing Chroma Feature Types for Automated Chord Recognition” in 42nd International Conference: Semantic Audio, 2011.
22. N. Jiang, “An Analysis of Automatic Chord Recognition Procedures for Music Recordings”, 2011.
23. M. Müller, S. Ewert, “CHROMA TOOLBOX: MATLAB IMPLEMENTATIONS FOR EXTRACTING VARIANTS OF CHROMA-BASED AUDIO FEATURES” in Proceedings of the12th International Conference on Music Information Retrieval (ISMIR) 2011. hal-00727791, versão 2 - 2012.
24. M. Marolt, “Transcription of polyphonic piano music with neural networks” in 10th Mediterranean Electrotechnical Conference, 2000, pp512-515.
25. M. Marolt, “a comparison of feed forward neural network architectures for piano music transcription”
26. Isophonics.[Online]: <http://isophonics.org/>. (Acessado em 10/11/2019).
27. B. Med, “Teoria da música”, 4ª ed. Brasilia-DF: musimed, 1996.
28. AudioLabs[Online]: <https://www.audiolabs-erlangen.de/resources/MIR/chromatoolbox/>. (Acessado em 16/11/2019).
29. mpi-inf[Online]: <http://resources.mpi-inf.mpg.de/MIR/chromatoolbox/>. (Acessado em 16/11/2019).
30. Github.[Online]: <https://github.com/livisghton/tcc>. (Acessado em 16/11/2019)
31. wikipedia. [Online]: <https://pt.wikipedia.org/wiki/Cifra_(música)>. (Acessado em 01/10/2019).
32. cifraclub. [Online]: <https://www.cifraclub.com.br/>. (Acessado em 01/10/2019)
33. cifras. [Online]: <https://www.cifras.com.br/>. (Acessado em 01/10/2019)
34. pegacifra. [Online]: <https://www.pegacifra.com.br/>. (Acessado em 01/10/2019)
35. F.Veloso, M.A.Feitosa. "O Ouvido Absoluto: bases neurocognitivas e perspectivas". 2013, p. 357 - 362.
36. P.J. dos Santos Junior. "Ouvido absoluto e ouvido relativo: sua natureza e relevância para a educação musical". In XVI Encontro Regional Sul da ABEM; 2014.
37. deeplearningbook. [Online]: <http://deeplearningbook.com.br/algoritmo-backpropagation-parte-2-treinamento-de-redes-neurais/>. (Acessado em 01/11/2019)

1. “Cifra club”. https://www.cifraclub.com.br/. Acessado em novembro de 2019 [↑](#footnote-ref-1)
2. “Cifras”. https://www.cifras.com.br/. Acessado em novembro de 2019 [↑](#footnote-ref-2)
3. “Pega cifra”. https://www.pegacifra.com.br/. Acessado em novembro de 2019 [↑](#footnote-ref-3)