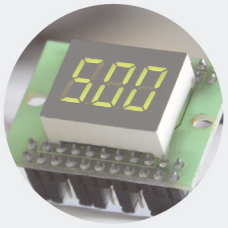
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES






# Separação de fontes de vocalizações ultrassónicas no estudo comportamental de murganhos

**VASCO JOSÉ ROSADO BATISTA FERNANDES MOTA**
novembro de 2024

POLITÉCNICO
DO PORTO

Politécnico do Porto
Instituto Superior de Engenharia do Porto

# Separação de fontes de vocalizações ultrassónicas no estudo comportamental de murganhos

**Vasco Mota**

Mestrado em Engenharia Electrotécnica e de Computadores
Área de Especialização em Automação e Sistemas


Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto

Novembro, 2024

*Esta dissertação satisfaz, parcialmente, os requisitos que constam da Ficha de Unidade Curricular de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Electrotécnica e de Computadores, Área de Especialização em Automação e Sistemas.*

**Candidato:** Vasco Mota, Nº 1191116, 1191116@isep.ipp.pt

**Orientação Científica:** Luís Coelho, LFC@isep.ipp.pt


Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
Rua Dr. António Bernardino de Almeida, 431, 4200–072 Porto

Novembro, 2024

# Agradecimentos

Ao Professor Doutor Luís Coelho, pelo apoio prestado e pela simpatia e paciência com que me atendeu.

À minha mãe e ao meu irmão que me acompanharam ao longo deste percurso, tanto nos bons como nos maus momentos.

Aos colegas e amigos que me ajudaram a encontrar formas de resolver alguns dos problemas enfrentados.

A todos aqueles que de alguma forma contribuíram para a realização deste trabalho.

# Resumo

A análise de vocalizações permite uma compreensão mais profunda do estado emocional do locutor, sendo influenciada pelas variações emocionais. Este fenómeno não ocorre apenas em humanos, o mesmo acontece noutros animais, como é o caso dos murganhos, *Mus musculus*, utilizados para estudar padrões psicológicos através das suas vocalizações. Devido às suas características, as comunicações que estabelecem são realizadas em frequências superiores a 20 kHz, sendo denominadas de *Ultrasonic Vocalizations* (USV).

Este projeto visa ajudar no estudo das USV de murganhos, quando não existem múltiplos microfones. A leitura com apenas um microfone faz com que todas as vocalizações estejam misturadas, levando a que todas as análises sejam mais demoradas e manuais. Desta forma, o projeto tem como objetivo ajudar em situações onde seja estudado um cenário entre progenitora e cria. Assim foi planeado e criado de um programa capaz de identificar os dois grupos de murganhos: a projenitora e as respetivas crias.

O programa criado teve como base a mistura de duas áreas da análise de vocalizações, a de USV de murganhos e a da diarização de locutor. Para tal, o programa teve de ser capaz de obter características das vocalizações e separá-las em grupos, utilizando um método de *clustering*. Desta forma, utilizou-se *Mel Frequency Cepstral Coefficients* (MFCC) para obter as características do sinal, que foram utilizados em dois métodos de *clustering*, o *Gaussian Mixture Models* (GMM) e K-means. No final, foi projetado um gráfico do espectrograma do áudio, onde se pode verificar os resultados de ambos os métodos.

Os resultados obtidos foram promissores, mas não definitivos. Os dois métodos experimentados tiveram resultados semelhantes, analisando corretamente por volta de 80% das vocalizações das crias, 75% das da progenitora e 60% em situações que estão os dois a comunicar.

Apesar do sucesso nos resultados, não deixa de ser necessário melhorias em projeções futuras, onde seja feita uma análise com mais dados, seja realizado um tratamento mais cuidado dos dados para a obtenção de características ou sejam abordados outros métodos de *clustering* que melhor se possam adequar, podendo passar por

métodos mais avançados como por *Deep Learning*.

**Palavras-Chave**: USV, murganhos, clustering, GMM, K-means, MFCC, sonogramas.

# Abstract

The analysis of vocalizations allows a deeper understanding of the speaker's emotional state, as it is influenced by emotional variations. This phenomenon does not occur only in humans; it is also observed in other animals, such as mice, Mus musculus, which are often used to study psychological patterns through their vocalizations. Due to their specific characteristics, their communications occur at frequencies above 20 kHz, known as *Ultrasonic Vocalizations* (USV).

This project aims to assist in the study of mice USV where multiple microphones are not available. Recording with only one microphone results in mixed vocalizations from different individuals, which makes all analyses more time-consuming and manual. Therefore, this project was designed to help with this analysis stage, specifically in scenarios involving a mother and her pups. As such, a program was planned and developed to identify the two groups of mice.

The program created is based by combining two areas of vocalization analysis: mouse USV and speaker diarization. For this, the program needed to be able to extract vocalization features and group them using a clustering method. Thus, *Mel Frequency Cepstral Coefficients* (MFCC) was used to extract signal features, which were then used in two clustering methods: *Gaussian Mixture Models* (GMM) and K-means. Finally, a spectrogram of the audio was plotted, where the results of both methods can be observed.

The results obtained were promising, but not definitive. Both methods produced similar results, correctly analyzing around 80% of the vocalizations made by the pups, also 75% that were made by the mother and only 60% of those that were made by both simultaneously.

Despite the success of the results, it's suggested that improvements should be done in future studies. Such work could include analyzing more data, refining data processing for feature extraction or exploring other clustering methods that may be more suitable, including advanced methods such as Deep Learning.

**Keywords**: USV, mice, clustering, GMM, K-means, MFCC, sonograms.

# Índice

# Lista de Figuras

# Lista de Tabelas

# Lista de Acrónimos

| | |
|---|---|
| **BIC** | *Bayesian Information Criterion* |
| **DCT** | *Discrete Cosine Transform* |
| **DFT** | *Discrete Fourier Transform* |
| **DNN** | *Deep Neural Network* |
| **EM** | *Expectation Maximization* |
| **FFT** | *Fast Fourier Transform* |
| **GB** | *Gigabyte* |
| **GiB** | *Gibibyte* |
| **GMM** | *Gaussian Mixture Models* |
| **HMM** | *Hidden Markov Models* |
| **IAs** | inteligências artificiais |
| **LSTM** | *Long Short-Term Memory* |
| **MFCC** | *Mel Frequency Cepstral Coefficients* |
| **PCA** | *Principal Component Analysis* |
| **RAM** | *Read Only Memory* |
| **SAD** | *Speech Activity Detection* |
| **SPL** | *Sound Pressure Level* |
| **STFT** | *Short-time Fourier Transform* |
| **USV** | *Ultrasonic Vocalizations* |

# Capítulo 1

# Introdução

A análise de vocalizações de animais, permite uma melhor compreensão de como eles agem e reagem em diferentes ocasiões. Desta forma, cientistas estudam os efeitos e comparam-nos com a resposta humana, levando a uma melhor comparação das duas partes. O murganho, denominado de *Mus musculus*, é um dos animais utilizado para estes testes e devido às suas características nas vocalizações, técnicas e materiais personalizados são essenciais.

## 1.1 Contextualização

O estudo das vocalizações de ratos tem ajudado investigadores a compreender melhor como não só estes reagem a diferentes estímulos, mas também os impactos que diversos tratamentos têm. Apesar de pouco intuitivo, estudos apontam que os resultados obtidos nos ratos, produzem resultados semelhantes para os seres humanos [1, 2, 3]. Desta forma, modelos de abuso de drogas, de depressão, de medo, de ansiedade, de doenças neurodegenerativas, de envelhecimento e de processamento de recompensa são estudados antecipadamente em ratos [4, 5], antes de serem estudados nos seres humanos. Assim, o processo de captação e análise dos ratos é essencial, realçando a importância da necessidade da qualidade dos instrumentos quer de *hardware*, quer de *software*.

As vocalizações destes roedores, encontram-se, maioritariamente, acima dos 20 kHz, ou seja, são inaudíveis para o ser humano. Apesar deste constrangimento, com a utilização de programas e aparelhagem específica, é possível analisar os ruídos

que produzem. Para além disso, os ratos possuem o gene FOXP2 que, ao contrário de muitos mamíferos,também se encontram nos seres humanos, sendo uns dos genes responsável pela fala [1]. As vocalizações dos ratos encontram-se entre as frequências de 22 kHz e de 50 kHz, podendo chegar a 115 kHz, onde frequências mais baixas têm uma conotação de infelicidade e as mais altas de felicidade [4, 5, 6, 7].

Analisando as vocalizações, as classificações destas variam de estudo para estudo e de programa para programa, podendo ser divididas em 15 grupos [7] ou até mesmo ultrapassando os 100 [8]. Deste modo, de forma a automatizar e agilizar o processo, várias inteligências artificiais (IAs)) foram criadas. Das diferentes IAss existem vários *softwares*, como por exemplo o MUPET [8] e o DeepSqueak [4]. Estes, usando metodologias diferente, analisam os dados e apresentam a respetiva categoria. O MUPET analisa através do processamento do sinal, enquanto o DeepSqueak processa através do meio visual.

## 1.2 Definição do Problema

Devido à falta de equipamento, apenas um áudio é obtido para analisar as vocalizações de cada teste. Desta forma, o programa utilizado até à data pela equipa de trabalho envolvida neste estudo, o DeepSqueak, torna-se ineficiente, visto que funciona melhor com o áudio previamente separado, normalmente obtido patravés de múltiplos microfones. Na situação atual, a catalogação é feita de forma manual com base nos ficheiros obtidos no programa, sendo um processo impreciso, trabalhoso e demorado.

### 1.2.1 Objetivos

Este projeto tem em vista:

- O desenvolvimento de um estado de arte onde são explorados *Ultrasonic Vocalizations* (USV) e métodos de análise;
- A criação de um protótipo capaz de distinguir as vocalizações da progenitora das vocalizações das crias;
- Avaliar os resultados obtidos pelo protótipo, para averiguar o seu funcionamento.

### 1.2.2 Resultados esperados

Com a conclusão do *software*, prevê-se que este seja capaz de visualizar o sinal e analisar o sinal, mostrando os momentos em que a progenitora ou as crias estão a comunicar. Esta visualização ocorrerá através de um gráfico temporal, variando entre 0, 1, 2 ou 3, (onde 0 representa que não está a haver comunicações, quando é 1

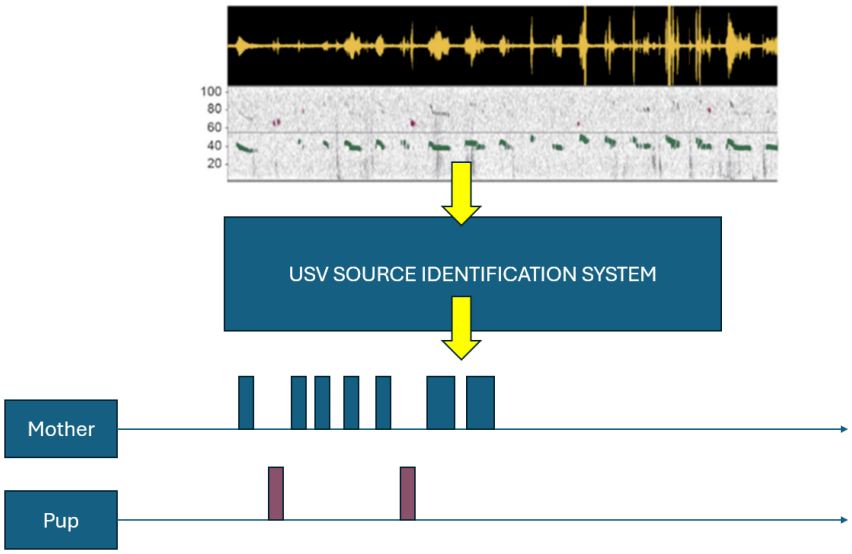



Figura 1.1: Evolução dos resultados esperados ao longo do projeto

apenas a progenitora está em comunicação, sendo 2 apenas as crias estão a produzir vocalizações e, por fim, é 3 nos momentos em que ambas, progenitora e crias, estão em comunicação) de forma a poder obter-se cada um dos seguintes resultados: não haver comunicações, apenas a progenitora comunicar, apenas as crias comunicarem ou estarem tanto as crias como a progenitora a comunicar. Este gráfico encontrar-se-á em conjunto com o espectro das comunicações, facilitando, assim, a observação de cada comunicação no seu contexto. Na Figura 1.1, encontram-se os vários passos que se prevê realizar para se alcançar o objetivo.

## 1.3 Plano de Trabalho

O projeto tem entrega prevista até ao dia 31 de outobro. Dessa forma, na Figura 1.2, encontra-se uma antevisão de como o projeto irá decorrer.

## 1.4 Organização da Dissertação

Esta dissertação está dividida em 6 capítulos:

- O Capítulo 1 tem por base dar uma breve contextualização do tema e expor o problema e uma possível resolução, mostrando também o plano de trabalho para a dissertação;
- No Capítulo 2 é feito um estudo dos vários projetos que tentam resolver problemas semelhantes ao do desta dissertação, salientando características específicas, de cada um dos trabalhos, considerados relevantes. Estes trabalhos são de duas vertentes: a análise de vocalizações e a diarização de locutor;

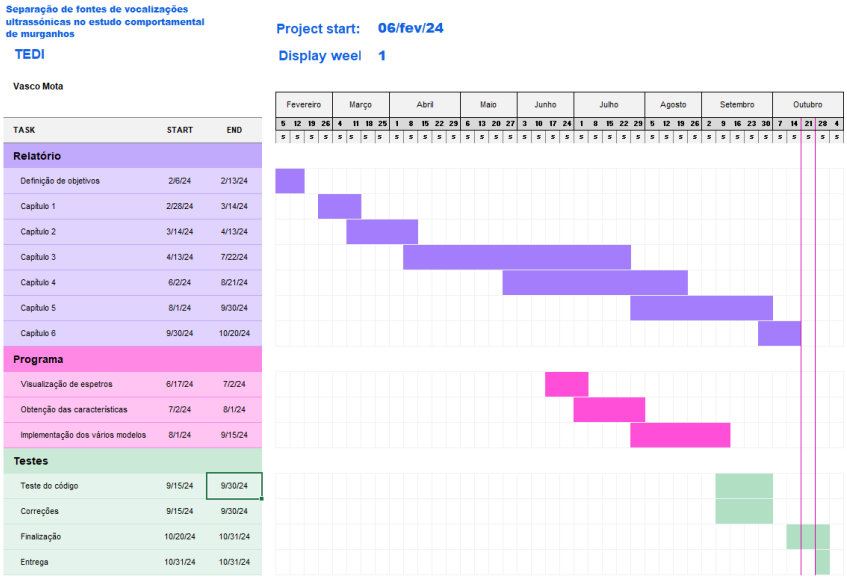



Figura 1.2: Plano previsto para o projeto

- O Capítulo 3 tem o intuito de explorar o processo de análise de USV e de diarização de locutor, como funcionam e como são criados. Para além disso, são referidos os vários modelos de tratamento de dados para se poder proceder à sua análise, desde a identificação de vocalizações à identificação de locutores;
- O Capítulo 4 explica a proposta de resolução, explorando os principais passos a realizar com base na revisão prévia. É também explorado o processo de desenvolvimento do projeto e da razão de cada decisão tomada até se chegar ao código final;
- O Capítulo 5 contém os resultados obtidos do programa, onde é explorado o significado desses resultados e tecidos comentários que os avaliam e os comparam;
- O Capítulo 6, o último capítulo, é composto por uma conclusão do resultado e do projeto como um todo. Também explora os próximos passos a dar para o melhoramento do programa.

# Capítulo 2

# Estado de Arte

Na análise de vocalizações de murganhos, em situações similares às estudadas, são utilizados vários microfones, de forma a poder distinguir-se rapidamente a progenitora das crias. No entanto, a existência de apenas um microfone impossibilita essa agilização, levando a que seja necessário uma diarização manual. Assim, torna-se necessário explorar não só que métodos são utilizados para a análise de USV, mas também métodos utilizados na área da diarização de locutor.

## 2.1 Análise de Padrões Acústicos das USV

Existem vários projetos que exploram a análise das comunicações entre murganhos. No entanto, o problema em estudo não é apenas uma análise de vocalizações, mas sim da identificação de quem produziu a USV. Desta forma, a diarização de locutor, é uma área que demonstra várias similaridades, pois esta tenta responder à questão: "Quem falou e quando?"[9]. Consequentemente, é de real importância averiguar que métodos são utilizados tanto para a análise de comunicação de murganhos como na diarização de locutor, visto que ambas as áreas estão interligadas para o sucesso do projeto.

## 2.2 Projetos de Análise de Comunicações de Murganhos

Para a análise de *Ultrasonic Vocalizations*, a utilização de sonogramas é fundamental no ramo, sendo aplicada numa grande variedade de projetos.

Browning *et al.* [2], utilizaram espectrogramas para fazer uma classificação manual das vocalizações. O objetivo do projeto era a investigação dos efeitos negativos causados por concussões. Para tal, devido à elevada quantidade de dados necessários para a averiguação, utilizaram os espectrogramas, obtidos por *Fast Fourier Transform* com comprimento de 512, para observar e catalogar manualmente. Posteriormente, através do *software* SASLabPro (Avisoft Bioacoustics, Alemanhã) fizeram uma classificação automática que, quando comprada com a manual, obteve uma identificação correta superior a 95% para vocalizações causadas por desgosto ou desconforto. Desta forma, a utilização do *software* era viável para a continuação do projeto.

Da mesma maneira, Wright *et al.* [7], também utilizaram sonogramas. Com o objetivo de identificar diferentes tipos de vocalizações emitidos por ratos a 50000 Hz e determinar que consequências a variação de anfetamina e de condições sociais causavam na quantidade e características das vocalizações. Os sonogramas foram obtidos com um FFT de comprimento 512, referindo ainda uma sobreposição de 75%, de onde fizeram as classificações. No entanto, não é referida qualquer ferramenta auxiliar para trabalhar as mais de 20000 USV.

Com o objetivo de verificar as alterações causadas pela mutação Foxp2, Chabout *et al.* [1] utilizaram o programa Mouse Song Analyzer v1.3 para fazer uma classificação automática das vocalizações. Este *software* utiliza as características do espectro do sinal são analisadas e com base nesses valores, são agrupadas em categorias, aquelas que não são identificadas, são catalogadas como sem classificação. No documento é referido que para a obtenção dos sonogramas foi utilizado uma FFT com comprimento de 256 e com sobreposição de 50%. Existe uma análise das vocalizações sem classificação, onde chegou-se à conclusão de que na sua maioria eram ruídos mecânicos, causados pelo rato, e pequenos ruídos que ele produziu que não eram chamamentos. Desta forma, esses dados não foram considerados para o projeto.

Ao contrário dos anteriores, Coffey *et al.* [4] exploraram as capacidades do programa DeepSqueak e a sua utilidade. Utilizando sonogramas numa rede neuronal, fizeram uma classificação automática das USV, permitindo ainda a utilização manual para aprimorar os dados. Os sonogramas foram gerados com base na vocalização e adaptados para a própria, variando valores como nfft, overlap e window, onde os dados obtidos foram utilizados para treinar uma rede neuronal Faster RCNN de deteção de objetos, otimizando a deteção de vocalizações.

Como o projeto de Coffey *et al.*, Zala *et al.* [10] focaram-se nas capacidades de um programa, no caso, o A-MUD. Neste projeto começaram por realizar uma *Short-time Fourier Transform* (STFT) para obter uma representação no domínio tempo-frequência, como numa FFT. O resultado foi um espectrograma que utilizou-se para identificar características como a largura de banda. O A-MUD foi também utilizado

para distinguir as vocalizações do ruído e identificar cada uma das vocalizações, que foram mostradas em espectrogramas. Para além disso, o algoritmo também fez uma análise da energia do sinal, fazendo correções para eliminar segmentos demasiado curtos ou demasiado longos, ou que estavam demasiado próximos uns dos outros. Por fim, foi concluído que quando comparado com leituras manuais do espectro, *software* foi capaz de identificar mais consistentemente as vocalizações quando comparadas a outro programa comercial, que não é identificado, mas aparenta ter características semelhantes ao Avisoft SAS LabPro (Avisoft Bioacoustics, Alemanha) e ao Sound Analysis Pro (SAP).

Burkett *et al.* [11] exploraram as capacidades do VoICE. No entanto, ao contrário dos outros projetos, não referiram em concreto o método utilizado para se obter o espectro, sendo apenas referido que foi utilizado o Matlab para realizar esse passo. Este pré processamento das vocalizações foi realizado de forma que as chamadas fossem extraídas automaticamente usando um limite de amplitude definido pelo utilizador. Posteriormente, foi realizado uma comparação dos contornos de frequência, tendo sido levado em consideração a correlação do tom e a sua variação, bem como a sobreposição temporal entre cada par de chamadas. Depois da análise foi concluído que o programa apresentou características vantajosas para a sua utilização na análise de vocalizações.

Para o estudo do programa MUPET, Segbroeck *et al.* [8] apresentam-no como um *software open source* de MatLab para a análise automática de vocalizações. Através da filtragem Gamatone e do Non-Negative Matrix Factorization são obtidas as características das vocalizações que são processadas pelo MUPET e por um método de *clustering*, o K-means. Terminaram o estudo com resultados positivos, chegando à conclusão de que o MUPET é capaz de analisar corretamente variações subtis nas sílabas.

## 2.3 Projetos de Diarização de Locutor

Para a diarização de locutor, existe uma grande variedade de projetos, tanto analisando de uma maneira teórica, explorando as várias tecnologias utilizadas, como de uma maneira prática, com a utilizações de técnicas novas ou melhoradas.

Para abordar o processo de diarização de locutor, explorando cada fase desse processo e mostrando a sua evolução, Anguera *et al.* [9], realizaram um estudo com o intuito de rever os vários métodos utilizados desde 2006 até à data da publicação, 2012. Durante o pré processamento do sinal, métodos como redução de ruído, Beamforming Acústico, *Mel Frequency Cepstral Coefficients* (MFCC) e *Speech Activity Detection* (SAD) foram ditos como amplamente utilizados. Para abordar a diarização, duas abordagens foram exploradas, Bottom-Up e Top-Down. Estas foram

acompanhadas por modelos como *Gaussian Mixture Models* (GMM) e *Hidden Markov Models* (HMM), modelos de aglomeração que utilizam MFCC e Beamforming Acústico, respetivamente. Information Bottleneck foi outra abordagem analisada no estudo, juntamente com as abordagens não paramétricas baseadas em modelos Bayesianos, Processos de Dirichlet e Modelos Hierárquicos de Dirichlet. Para além disso, é referido o desenvolvimento de técnicas de obtenção de características como as prosódicas, onde podem ser acrescentados ao pré processamento para métodos como MFCC.

Com o objetivo de melhorar a eficácia dos MFCC para a diarização de locutor, Friedland *et al.* [12] realizaram um estudo utilizando algumas características prosódicas, juntamente com algoritmos de SAD, para acompanhar os MFCC. Salientaram dois modelos para a diarização, o GMM e o *Bayesian Information Criterion* que foram utilizados em simultâneo de forma a complementar os resultados de cada. No final, é concluído o sucesso dos melhoramentos dos resultados.

Elizabeth Shriberg [13], teve o intuito de explorar a utilização de características de alto nível em sistemas de reconhecimento automático de locutor. Argumentou que sistemas baseados em GMM, apesar de dominantes e eficazes, não conseguem capturar informações linguísticas e de longo alcance que residem no sinal de fala. Ao explorar o conceito de características de alto nível e as suas vantagens, concluiu que apesar de mais impreciso quando comparado individualmente com métodos como o GMM, quando utilizado em conjunto, existe um melhoramento na precisão significativo. Realçou a utilização de características como os MFCC e características prosódicas para a análise dos dados em modelos GMM, como já referido, mas também em modelos HMM, Dynamic Time Warping e de contorno de tom.

O projeto, de Shum *et al.* [14], teve como objetivo melhorar a precisão da diarização de locutor, onde fosse possível associar diferentes segmentos de fala pertencentes ao mesmo orador. Através de uma abordagem de Variabilidade Total foi explorado a variabilidade de intra-conversação, utilizando PCA para comparar *i-vectors*, vetores de baixa dimensão obtidos por MFCC. Este método permitiu uma melhor separação dos *i-vectors*, de forma a poderem ser agrupados pelo algoritmo de *clustering*, K-means, de forma a separar cada vetor para os locutores correspondentes. Os resultados obtidos foram favoráveis, tendo alcançado uma taxa de erro de diarização de 0,9% em situações de conversa de telemóvel.

Como no projeto retratado por Shum *et al.*, Dehak *et al.* [15] teve o objetivo de melhorar a precisão da diarização de oradores, especificamente, em cenários onde não se sabe a quantidade de locutores. Abordando o modelo de GMM com base num tratamento prévio utilizando o PCA para reduzir *i-vectors*, obtidos por uma análise fatorial. Com o *clustering* por GMM definindo o que cada orador dizia, utilizou-se um algoritmo de ressegmentação baseado em Viterbi onde os vários segmentos foram refinados para cada locutor. No final, a abordagem teve resultados comparáveis aos

de um *benchmark* de um CallHome.

Horiguchi *et al.* [16] propôs um novo método de diarização. Utilizando um modelo End-to-End, o projeto diarizou os locutores independentemente de quantos se encontravam presentes no áudio, para tal o modelo foi denominado de End-to-End Neural Diarization with Encoder-Decoder based Attractors. Utilizando codificadores Transformer empilhados, para extrair *embeddings* de áudio, convertem a sequência de características acústicas de entrada numa sequência de *embeddings*. Calculando atratores, tanto a nível local como a nível global, o modelo foi capaz de lidar com gravações com um grande número de oradores, dividindo a tarefa em problemas menores e mais fáceis de gerir. Para a realização de *clustering* foi escolhido um método não supervisionado para encontrar correspondências. Para além disso, na situação de a quantidade de locutores ser considerada abaixo do estimado, foi utilizado os atratores globais e para a situação contrária os locais, para corrigir esse valor. O *clustering* foi comparado com um baseado em vetores-x, o state-of-the-art x-vector-based method, métodos de End-to-End implementados, obtendo um melhores resultados em termos de taxa de erro de diarização.

Num trabalho mais recente, Horiguchi *et al.* [17], teve o intuito de melhorar a rede neural End-to-End, que se tinha desenvolvido [16]. Exploram as desvantagens de que a rede neural apresenta, referindo, por exemplo, a dificuldade em cenários onde o número de locutores varia. Para resolver o problema, utilizaram um atrator encoder-decoder-based, com o método de *Long Short-Term Memory* (LSTM). Desta forma, a *performance* foi melhorada, mas alguns problemas ainda se mantiveram, como o número limite de falantes, que é definido pelo *set* durante o processo de treino.

## 2.4 Lista de Projetos

Na Tabela 2.1 encontra-se uma lista simplificada de vários projetos de análise de vocalizações de murganhos, é referido o objetivo, método de obtenção de características e o modelo utilizado.

Por outro lado, na Tabela 2.2 encontra-se uma lista simplificada de vários projetos de diarização de locutor, é também referido o objetivo, método de obtenção de características e o modelo utilizado.

## 2.5 Resumo

No capítulo 2 foi realizado um estudo de vários projetos, analisando os métodos utilizados, para se perceber como enfrentar o problema deste projeto.

Começando o capítulo por uma análise de USV, na secção 2.1, foi, posteriormente, continuado por uma análise de dois problemas distintos: a análise de comunicação de murganhos e a diarização de locutor. Assim, na secção 2.2, foi realizado um estudo de projetos de análise de comunicações em USV e, posteriormente, projetos de diarização de locutor. Desta forma, observou-se como é que eram obtidas as características das vocalizações para ambos os temas, como por exemplo, o método de MFCC e os *embeddings* (utilizados nos modelos mais avançados), exclusivos na diarização de locutor, e, por fim, o modelo utilizado, onde na sua maioria era de *clustering*.

Por fim, terminou-se o capítulo com duas tabelas que contêm um resumo de vários projetos, de forma a simplificar o que cada um deles utilizava.

Tabela 2.1: Resumo de Projetos para a análise de comunicações de murganhos

| Projetos | Objetivos | Características e Embeddings | Modelo/Programa |
|---|---|---|---|
| [2] | Investigação dos efeitos negativos causados por concussões | Sonograma por FFT | SASLabPro |
| [7] | Identificação de diferentes tipos de vocalizações e suas causas | Sonograma por FFT | Não referido |
| [1] | Verificação das alterações causadas pela mutação Foxp2 | Sonograma por FFT | Mouse Song Analyzer v1.3 |
| [4] | Exploração das funcionalidade do programa DeepSqueak para a análise de USV | Sonograma (método não referido) | DeepSqueak |
| [10] | Exploração das funcionalidade do programa A-Mud para a análise de USV | Sonograma por STFT | A-MUD |
| [11] | Exploração das funcionalidade do programa VoICE para a análise de USV | Sonograma (método não referido) | VoICE |
| [8] | Apresentar o MUPET como uma nova ferramenta para a análise de USV de ratos | Filtragem Gammatone e Non-Negative Matrix Factorization | MUPET e K-means |

Tabela 2.2: Resumo de Projetos de diarização de locutor

| Projetos | Objetivos | Características e Embeddings | Modelo/Programa |
|---|---|---|---|
| [9] | Estudo de técnicas de diarização | Redução de ruído, Beamforming Acústico, MFCC e SAD | GMM, HMM, Information Bottleneck, Processos de Dirichlet e Modelos Hierárquicos de Dirichlet |
| [12] | Melhorar a eficácia de MFCC | MFCC, características prosódicas e SAD | GMM e BIC |
| [13] | Explorar a utilização de características de alto nível | MFCC e características prosódicas | GMM, HMM, Dynamic Time Warping e contorno de tom |
| [14] | Melhorar a precisão da diarização de locutor | MFCC, vetor-i | PCA e K-means |
| [15] | Melhorar a precisão da diarização de oradores | Análise Fatorial | GMM, PCA e ressegmentação baseada em Viterbi |
| [16] | Novo método de diarização | Vetores-x e atratores | *Clustering* não supervisionado, End-to-End, Codificadores Transformer |
| [17] | Melhorar a diarização de locutores | Atratores, Vetores-i e vetores-x | LSTM, Codificadores Transformer, Agglomerative Hierarchical Clustering, Variational Bayes |

# Capítulo 3

# Revisão de Literatura

Existem vários métodos de obtenção de características e vários modelos para a sua utilização, no entanto, nem todos funcionam de igual modo, nem têm por base os mesmos processos. Desta forma, foi necessário analisar o que os distingue, bem como o processo para a obtenção não só das características, mas também do resultado final dos modelos.

## 3.1 Vocalizações Ultrassónicas

Uma *Ultrasonic Vocalizations* (USV), é um tipo de vocalização inaudível para o ser humano, visto encontrar-se na gama dos ultrassons, frequências acima dos 20 kHz [18]. Murganhos, como referido na secção 1.1, são um exemplo de animais que fazem todas as suas comunicações inaudíveis para os cientistas que os estudam. No entanto, o estudo destes pequenos roedores é essencial para uma melhor compreensão de como os efeitos sociais influenciam [7, 19], drogas [3], ferimentos [2] ou acasalamento [6, 20].

O estudo de USVs em roedores permite uma melhor compreensão de reações, visto que, ao contrário do ser humano, estes animais não apresentam movimentos característicos quando vocalizam, por exemplo, durante o acasalamento, o cortejo é feito maioritariamente pelo macho, através de vocalizações [18, 20].

Para além disso, a frequência de uma vocalização, permite identificar um comportamento agressivo ou amistoso. Vocalizações perto dos 22 kHz são comuns como uma reação quando um predador está presente ou quando existe algum conflito.

Estas vocalizações podem causar a paralisação temporária do recetor, por provocar medo e ansiedade, levando a um aumento de atividade neurológica na área de amígdala cerebelosa. Nas comunicações perto das frequências de 50 kHz, existe um efeito contrário para o recetor, existe uma redução da atividade na área da amígdala cerebelosa e há um aumento na atividade dos núcleos accumbens, uma área que está ligada ao processamento de recompensas [21, 22].

Em relação à utilização de drogas, as vocalizações estavam mais compreendidas nas frequências de 22 kHz, quando a droga era restrita, mas quando esta era mais acessível, as vocalizações já se encontravam mais frequentemente nas frequências de 50 kHz [3]. Para além disso, durante o consumo, o formato das vocalizações também demonstra ser alterado [7].

Por último, outra aplicabilidade do estudo das vocalizações vem do facto das progenitoras comunicarem, através de chamamentos, com as crias. Desta forma, é possível estudar os comportamentos quando estão afastadas e posteriormente são reunidas [11].

## 3.2 Características acústicas

Qualquer onda contém características que a distingue de outras ondas. Quando se ouve duas pessoas a falar, facilmentee apenas através da audição, é possível perceber quem está a falar. No entanto, mesmo nessas situações, certas características são inerentes à língua e pronúncia [23], ou seja, uma vogal, quando analisada, tem características semelhantes quando falada por duas pessoas diferentes, com a mesma pronúncia. A obtenção destas características é essencial em diversos estudos [24, 25, 26] e o mesmo se aplica para a análise de comunicações de murganhos, onde ao contrário da comunicação entre humanos, esta é inaudível para seres humanos, por ter frequências acima do limite da audição. Assim, qualquer comunicação tem de ser analisada em computador com base nas características acústicas. Alguns destes métodos são também utilizados na obtenção das características da voz de uma pessoa.

As vocalizações de um rato, podem ter várias formas como as presentes na Figura 3.1, variando de frequência, amplitude, duração e complexidade [18], sendo tratadas com base em sonogramas [1, 2, 4, 7]. Por outro lado, para a análise de comunicações, mais especificamente a diarização de locutor, métodos como MFCC [9] ou a obtenção de características prosódicas são frequentemente utilizados [9].

### 3.2.1 Sonogramas

Sonogramas são utilizados para a obtenção de gráficos da amplitude das frequências ao longo do tempo. São espectrogramas caracterizados por se referirem apenas a

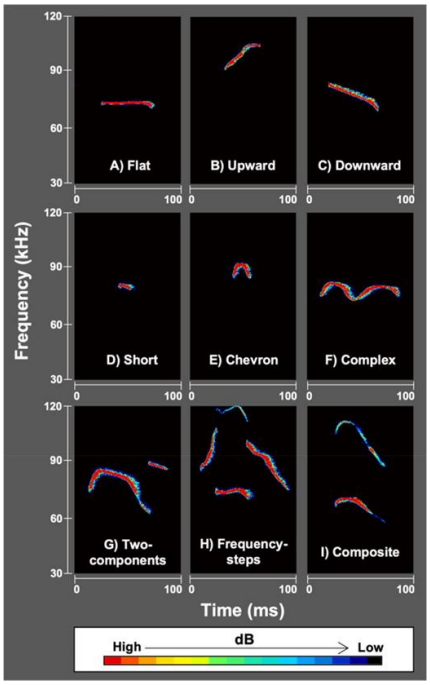



Figura 3.1: Exemplos de tipos de vocalizações de um rato [18]

sons. A principal função é a possibilidade de se observar a amplitude das frequências dos vários sons, isto é, qualquer som tem intrinsecamente uma frequência que pode ter maior ou menor intensidade. Deste modo, no gráfico podem analisar-se as frequências e a sua variação ao longo do tempo. Na Figura 3.2 encontra-se presente um exemplo de um sonograma. É possível analisar através do mapa de cores que nas regiões das manchas amarelas encontram-se as vocalizações com maior amplitude. Ao analisar um período de tempo, é possível observar a sua variação, mostrando, assim, o som.

Para se chegar a este tipo de gráficos é necessário executar determinados passos:

- Separar em janelas;
- Aplicar um algoritmo de suavização;
- Aplicar uma Transformada de Fourier;
- Calcular a magnitude.

### Janelas

Primeiro é realizada uma separação em segmentos, chamados "janelas". A duração de cadajanela é adaptada para o sinal correspondente, podendo ser inferior a 1 ms quando se trata de USV[10], Pode haver sobreposição de janelas e quanto maior esta for, maior será a precisão mas mais tempo demorará e recursos usará. Por isso, o valor de sobreposição é algo que varia de projeto para projeto [1, 10].

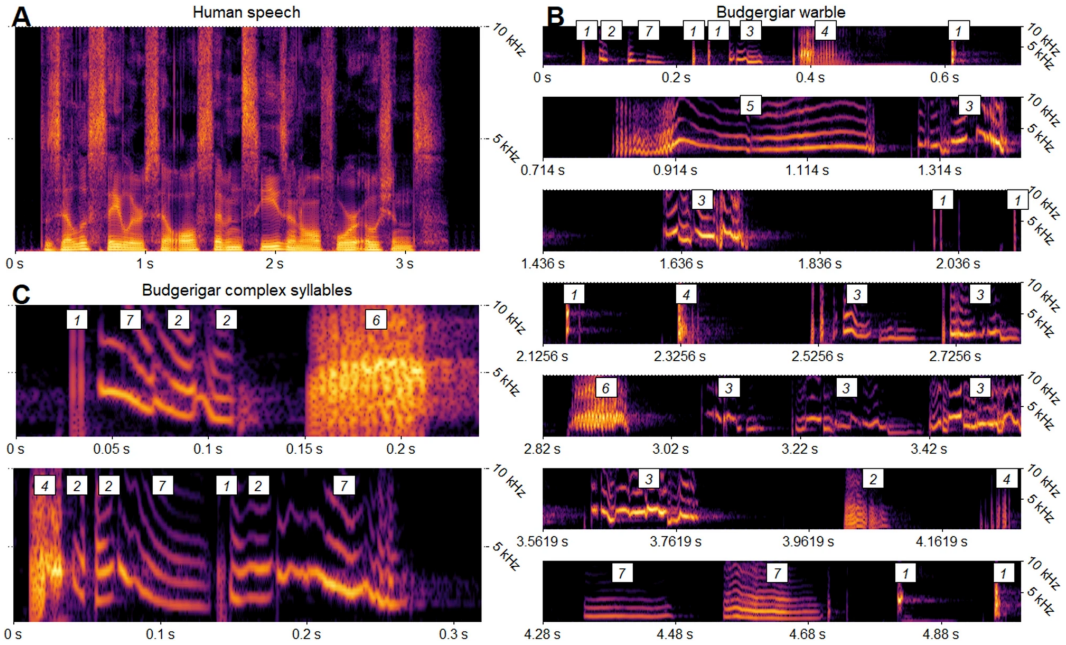



Figura 3.2: Exemplos de sonogramas [27]

### Suavização

Devido à possibilidade de distorção causada pela criação de janelas, é aplicado um algoritmo de suavização, sendo mais usuais o algoritmo de Hamming [2] e o de Hanning [10]. Estes dois algoritmos são diferenciados pelas constantes que contêm.

A equação de Hamming é definida por [28]:

$$w(n) = 0,54 - 0,46 \times cos(\frac{2\pi n}{N-1}) \tag{3.1}$$

Por outro lado, a equação de Hanning é definida por:

$$w(n) = \frac{1}{2} - \frac{1}{2}cos(\frac{2\pi n}{N-1}) \tag{3.2}$$

Onde:

- $w(n)$ é o valor da janela no ponto $n$;
- $N$ é o número total de amostras na janela:
- $n$ varia de 0 a $N-1$.

A janela de Hamming inicia com um valor próximo de 0, e vai aumentando até atingir o centro da janela. Nesta altura, o valor fica perto de 1. Após esse ponto, o valor da janela diminui novamente para 0, ao terminar. Ao multiplicar o sinal por essa janela, as amostras localizadas nas bordas da janela são suavizadas, o que resulta na redução de descontinuidades abruptas.

Para o algoritmo de Hanning, também começa e termina com valores iguais a 0, com um valor máximo de 1 no meio da janela. Contudo, a transição das bordas para o centro é mais suave quando comparada à janela de Hamming. Ao aplicar a multiplicação do sinal pela janela de Hanning, o resultado é uma atenuação mais suave nas bordas, em contraste com o outro algoritmo.

Desta forma, a janela de Hanning tem mais intensidade ao atenuar as bordas da janela, reduzindo as perdas espectrais. No entanto, pode diminuir a resolução em frequência. A janela de Hamming, por seu lado, tem um melhor controlo sobre as perdas espectrais, pois as bordas não ficam em 0, mas sim a um valor muito próximo deste.

### Transformada de Fourier

Para obter o espectro, é necessário fazer uma Transformada de Fourier, pois converte o sinal para o domínio das frequências. Para tal, utiliza-se a equação [29]:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt \tag{3.3}$$

Onde:

- $X(f)$ é o sinal transformado no domínio da frequência;
- $x(t)$ é o sinal no domínio do tempo;
- $f$ é a frequência em Hertz (Hz);
- $j$ é a unidade imaginária;
- $e^{-j2\pi ft}$ é uma função complexa que decompõe o sinal em suas componentes sinusoidais.

No entanto, para o caso do sonograma, como é um sinal digital, amostrado em intervalos de tempo discretos, utiliza-se a *Discrete Fourier Transform* (DFT). A sua equação é [30]:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi kn}{N}} \tag{3.4}$$

Onde:

- $X(k)$ é o valor da frequência $k$ no espectro (para $k = 0, ..., N-1$);
- $x(n)$ é o valor do sinal no tempo (para $n = 0, ..., N-1$);
- $N$ é o número total de amostras;
- $k$ é o índice associado a uma frequência $f_k = \frac{k}{NT}$, onde $T$ é o intervalo de tempo entre as amostras;

- $j$ é a unidade imaginária;
- $e^{\frac{-j2\pi kn}{N}}$ é uma função complexa que decompõe o sinal em suas componentes sinusoidais.

Em alguns projetos é referida a utilização de *Fast Fourier Transform* (FFT) ou de *Short-time Fourier Transform* (STFT). A primeira, é um algoritmo eficiente que calcula a DFT.A STFT é uma extensão da FFT, para sinais cujas propriedades de frequência variam ao longo do tempo. Isso é útil quando se pretende saber não apenas que frequências estão presentes num sinal, mas quando elas ocorrem. A sua equação é definida por [31]:

$$X(t, f) = \int_{-\infty}^{\infty} x(\tau)h^*(\tau - t)e^{-j2\pi ft} \tag{3.5}$$

Onde:

- $X(t, f)$ é o resultado, representando o conteúdo de frequência $f$ no momento $t$;
- $x(\tau)$ é o sinal no tempo;
- $h^*(\tau - t)$ é a função que limita o intervalo de tempo a ser analisado;
- $j$ é a unidade imaginária;
- $e^{-j2\pi ft}$ é uma função complexa que decompõe o sinal em suas componentes sinusoidais.

**Magnitude**

Após calcular $X$, é necessário calcular a magnitude. Como $X$ é um número complexo definido por parte real e parte imaginária, $a + jb$, a sua magnitude é calculada pelo seu módulo:

$$|X(k)| = \sqrt{a^2 + b^2} \tag{3.6}$$

Onde:

- $|X(k)|$ é o módulo de $X$;
- $a$ é a componente real;
- $b$ é a componente imaginária.

### 3.2.2 MFCC

*Mel Frequency Cepstral Coefficients* (MFCC) é um método utilizado em vários projetos de diarização de locutor. Este método obtém coeficientes referentes ao sinal a partir dos quais pode ser melhor trabalhado. Este é particularmente útil na diarização de locutor devido à sua capacidade de capturar as propriedades acústicas das palavras, isto é, ajuda a distinguir diferentes fonemas e palavras, independentemente do locutor, possibilitando a sua interpretação [9, 14].

O seu processo é dividido pelas seguintes fases [32]:

1. São criadas janelas;
2. É suavizado, normalmente tanto pela função ou de Hanning ou de Hamming, dependendo do objetivo;
3. Através da *Fast Fourier Transform* (FFT) é realizada uma computação espectral de cada segmento;
4. É aplicado o Mel Filter Bank;
5. É feito uma compressão logarítmica em cada frequência;
6. É aplicado um *Discrete Cosine Transform* (DCT) para se obter os coeficientes.

Como é possível observar, alguns dos passos são os mesmos que os descritos para os sonogramas na secção 3.2.1, como a separação do áudio em segmentos, a suavização e a aplicação da FFT.

Antes de explorar os três pontos restantes, é importante salientar que os MFCC apresentam um problema crucial: deveriam ignorar o que está a ser dito, o que não acontece, pois eles capturam informações sobre o conteúdo, podendo levar a inconsistências no reconhecimento de locutor. Por sua vez, para um processo de reconhecimento de fala, as características vocais do locutor também são capturadas, o que torna possível a existência de erros sobre o que foi dito [9].

Os coeficientes obtidos neste método são utilizados em algoritmos como *Gaussian Mixture Models* (GMM) [33, 15], K-means [33], vetor-i [17, 34] ou vetor-x [16, 17, 34].

#### Mel Filter Bank

Através do Mel Filter Bank é realizada a separação das frequências em bandas, em filtros triangulares [35, 36, 37]. Na Figura 3.3 é possível observar o efeito utilizado, salienta-se que o número de frequências é definido antes do processo.

Este filtro é computado utilizando o sistema:

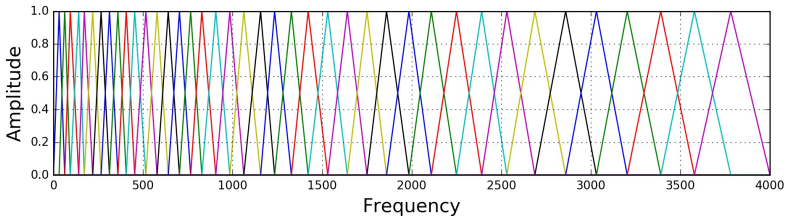



Figura 3.3: Mel Filter Bank [35]

$$H_m(k) = \begin{cases} 0 & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \leq k < f(m) \\ 1 & k = f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)} & f(m) < k \leq f(m+1) \\ 0 & k > f(m+1) \end{cases} \tag{3.7}$$

Onde:

- $f(m)$ é a frequência central do filtro triangular
- $\sum_m^{M-1} H_m(k) = 1$
- $k$ é o índice da frequência

No entanto, antes de se aplicar o Mel Filter Bank é feito um mapeamento do espectro. Para tal utiliza-se a Mel Scale. Esta é computada através da equação:

$$m = 2595 log_{10}(1 + \frac{f}{700}) \tag{3.8}$$

E a sua inversa é definida por:

$$f = 700(\frac{m}{2595}) \tag{3.9}$$

Onde:

- $m$ frequência na escala Mel
- $f$ frequência em Hertz (Hz)

### Compressão Logarítmica

Depois de se filtrar, é necessário fazer uma compressão logarítmica na qual, calcula-se a energia com a equação [37]:

$$E_i = \sum_{k=f(m-1)}^{f(m+1)} |X(k)|^2 \times H_m(k) \tag{3.10}$$

Onde:

- $f(m)$ é a frequência central do filtro triangular;
- $k$ é o índice da frequência;
- $H_m$ é o resultado após o Mel Filter Bank;
- $X(k)$ é o sinal após a DFT.

A compressão logarítmica é, por fim, calculada com:

$$E'_i = log_{10}(E_i) \tag{3.11}$$

Onde:

- $E_i$ é ó valor da energia de cada banda de frequência $i$ do Mel Filter Bank.

***Discrete Cosine Transform***

Finalmente, é computado os coeficientes cepstrais. Estes são calculados com a equação de DCT [36, 37]:

$$c(k) = \sum_{n=0}^{N-1} log_{10}(E_n) \times cos(\pi k \frac{2n-1}{2N}), \quad 0 \leqslant k < N \tag{3.12}$$

Onde:

- $log_{10}(E_n)$ são as energias logarítmicas de cada banda;
- $k$ é o índice do coeficiente cepstral;
- $N$ é o número total de bandas do Mel Filter Bank;
- $n$ é índice da banda atua.

### 3.2.3 Características Prosódicas

As características prosódicas referem-se a aspetos da fala que estão ligados à entoação, ritmo, duração e intensidade da voz. Dessa forma, elas concentram-se em como as palavras são pronunciadas e nas variações que podem transmitir informações semânticas e emocionais. Estas características são utilizadas para complementar outros métodos como o MFCC, que como descrito, pode levar a inconsistências no reconhecimento de locutor ou no que foi dito, podendo levar a um melhoramento de 30% quando utilizado em simultâneo com o MFCC[9].

Dependendo da fonte, o número de características varia, podendo chegar às 70 [9]. Na Tabela 3.1 observa-se uma lista que contém 52 possíveis características a serem utilizadas [12].

Tabela 3.1: Exemplo de uma lista de características prosódicas [12]

| rank | feature category | feature | $\frac{between}{within}$ |
|---|---|---|---|
| 1 | pitch | f0_median | 26.375387 |
| 2 | pitch | f0_mean | 9.412929 |
| 3 | formants | f4_stdev | 3.356307 |
| 4 | pitch | f0_min | 3.072750 |
| 5 | formants | f4_min | 2.885247 |
| 6 | harmonics-to-noise ratio | h2n_mean | 2.845387 |
| 7 | formants | f4_mean | 2.615583 |
| 8 | formants | f5_mean | 2.437822 |
| 9 | long-term average spectrum | ltas_stdev | 2.322184 |
| 10 | formants | f5_stdev | 2.251312 |
| 11 | formants | f4_median | 2.191332 |
| 12 | formants | f5_min | 2.170989 |
| 13 | formants | min_disp | 2.058759 |
| 14 | formants | f5_median | 2.043392 |
| 15 | energy | en_max | 1.952964 |
| 16 | formants | f3_stdev | 1.517770 |
| 17 | formants | mean_disp | 1.516091 |
| 18 | formants | f3_min | 1.356388 |
| 19 | formants | f3_mean | 1.236645 |
| 20 | long-term average spectrum | ltas_slope | 0.987187 |
| 21 | formants | f5_max | 0.939182 |
| 22 | formants | f3_median | 0.934186 |
| 23 | formants | f4_max | 0.839959 |
| 24 | formants | f3_max | 0.750493 |
| 26 | harmonics-to-noise ratio | h2n_stdev | 0.691567 |
| 27 | formants | f1_median | 0.597408 |
| 28 | formants | f1_mean | 0.553273 |
| 29 | formants | f1_min | 0.545544 |
| 30 | long-term average spectrum | ltas_fmin | 0.507023 |
| 31 | formants | f2_stdev | 0.498375 |
| 33 | formants | f2_median | 0.419127 |
| 34 | formants | f2_max | 0.409044 |
| 35 | harmonics-to-noise ratio | h2n_diff | 0.403711 |
| 36 | pitch | f0_max | 0.365185 |
| 37 | long-term average spectrum | ltas_lph | 0.361591 |
| 38 | harmonics-to-noise ratio | h2n_max | 0.351710 |
| 39 | formants | f2_mean | 0.349525 |
| 40 | formants | f2_min | 0.342969 |
| 41 | formants | max_disp | 0.281981 |
| 42 | pitch | f0_stdev | 0.225521 |
| 43 | pitch | f0_diff | 0.222980 |
| 44 | formants | f1_max | 0.200953 |
| 46 | formants | f1_stdev | 0.189932 |
| 47 | energy | en_avg | 0.162019 |
| 48 | long-term average spectrum | ltas_fmax | 0.086992 |
| 49 | energy | en_stdev | 0.066931 |
| 50 | energy | en_mean | 0.062653 |
| 51 | energy | en_min | 0.059033 |
| 52 | energy | en_diff | 0.056007 |

No entanto, para a diarização de locutor não são todas consideradas. Assim, as características mais utilizadas são [12]:

- Tom;
- Energia;
- Formato;
- Relação Harmônicos-Ruído;
- Espectro Médio de Longo Termo.

### Tom

O tom falado por alguém é característico de cada pessoa, sendo definido por diferenças anatómicas, como o comprimento ou a massa das pregas vocais, na laringe.

Uma das principais causas é o sexo e a idade de uma pessoa, havendo, por norma, claras diferenças quando se compara a voz de um homem com a de uma mulher ou criança.

O tom pode ser extraído através de programas como o PRAAT, da qual, posteriormente pode ser utilizado para se calcular:

- Média;
- Mediana;
- Mínimo e máximo (Valores que representam os extremos da distribuição, percentis 5 e 95);
- Amplitude da variação da frequência fundamental;
- Desvio Padrão;
- *Swoj* (Inclinação da curva de tom).

**Energia**

A energia de uma vocalização, a sua intensidade ou "o quão 'alto' está", não é uma característica causada pelas anatomias de uma pessoa, mas sim pelo ambiente que a rodeia ou o seu estado emocional. Deste modo, a energia tem a capacidade de diferenciar falantes.

Para medir a energia é necessário tratar os dados para uma medida de intensidade como os dB (*Sound Pressure Level* (SPL)), obtidos pela equação:

$$L = 20log_{10}(\frac{p_i}{p_0}) \tag{3.13}$$

Onde:

- $L$ é o nível de pressão sonora em decibéis;
- $p_i$ é a pressão sonora medida na amostra $i$;
- $p_0$ é a pressão sonora de referência.

Com estes valores é possível obter as suas características, sendo a maioria as mesmas que as do tom:

- Média;
- Amplitude;
- Mínimo e máximo;
- Desvio padrão.

### Formato

Influenciados pela configuração e dimensões da cavidade oral e nasal, os formantes são componentes característico da fala, referem-se a picos de energia acústica que ocorrem em frequências específicas, aproximadamente a cada 1000 Hz. Os formantes são utilizados para a identificação de vogais e alguns sons consonânticos, mas duas pessoas, devido às diferenças anatómicas, produzem formantes muito próprios, contribuindo para a identificação de locutor.

Os formantes podem ser obtidos através de algoritmos de análise espectral, onde para cada formante são calculadas estatísticas semelhantes às anteriores, média, mediana, mínimo, máximo e desvio padrão. Além disso, a soma das diferenças (mínimo, máximo e média) entre formantes consecutivos indica a dispersão dos mesmos.

### Relação Harmónicos-Ruído

A relação harmónicos-ruído quantifica o ruído que foi adicionado no sinal vocal, sendo a proporção entre a energia harmónica e a energia ruidosa no sinal da voz. Definida em dB, permite identificar características na voz, como uma voz rouca, caracterizada por um valor mais baixo, ou clara, caracterizada por um valor mais alto.

Este valor é obtido pela equação:

$$H = 10log_{10}(\frac{energia\ harmónica}{energia\ ruidosa}) \tag{3.14}$$

Onde:

- $H$ é o valor da relação em dB;
- $energiaharmónica$ é a energia associada à periodicidade da vibração das cordas vocais em percentagem;
- $energiaruidosa$ é a energia associada a irregularidades na vibração ou turbulência no fluxo de ar em percentagem.

Com estes dados, as mesmas estatísticas são aplicadas: média, mínimo, máximo, amplitude e desvio padrão.

### Espectro Médio de Longo Termo

O espectro médio de longo termo é uma representação da distribuição média da energia espectral ao longo de um segmento de fala relativamente longo, sendo capaz de revelar alguns padrões do locutor. Novamente, estatísticas como média, mínimos, máximos, amplitude, desvio padrão e a inclinação podem ser retiradas para se utilizar na diarização de locutor.

## 3.3 *Embeddings*

Alguns projetos utilizam *embeddings* para posteriormente serem analisados, quer por modelos de *clustering* quer por modelos de *machine learning* e *deep learning*.

### 3.3.1 *I-vectors*

Os *i-vectors*, também chamados por vetores de identidade, são representações vetoriais de baixa dimensionalidade que compacta as características de um locutor. Derivados de estatísticas acústicas, como a de Análise de Variabilidade Total, um tipo de Análise Fatorial Probabilística, os *i-vectors* são utilizados em tarefas de reconhecimento e diarização de locutor. Este tipo de análises modela a variabilidade inter-locutor e intra-locutor, de forma a que se capture características distintas de um locutor [33, 38, 39].

O primeiro passo é a obtenção de características, como por exemplo o MFCC. Com estes coeficientes, calculam-se as estatísticas de Baum-Welch que representam a probabilidade de cada segmento de fala pertencer a cada componente de uma Mistura Gaussiana de Modelo de Fundo Universal, um GMM combinado com um Modelo de Fundo Universal [40]. Por fim, feita uma análise fatorial probabilística, onde se obtém o *i-vector*. A equação geral do *i-vector* é dada por:

$$M = m + Tw \tag{3.15}$$

Onde:

- $M$ é o supervetor de médias do locutor;
- $m$ é o supervetor de médias do GMM-UBM;
- $T$ é a matriz de variabilidade total, que modela as variações inter-locutor e intra-locutor;
- $w$ é o vetor-i, que representa as características do locutor.

Os *i-vectors* computados, podem ser agrupados utilizando métodos de clustering, como o K-means, para se identificar os locutores. Podem também ser utilizados em modelos de Two-Stage Voice Activity Detection [41].

### 3.3.2 *X-vectors*

Os *x-vectors* surgem de *Deep Neural Network* (DNN) sendo semelhantes aos *i-vectors*, mas derivados e utilizados de formas diferentes. Para obter os *x-vectors*, é necessário processar segmentos de falas por uma DNN treinada, que gera um vetor de características que representa a identidade do locutor, esse vetor é denominado de *x-vector* [16, 17, 42]. O primeiro passo para é obter as características do sinal, através de

MFCC, por exemplo. Estas características são utilizadas no modelo DNN, como o ResNet101.

O ResNet101 é uma rede neural convolucional profunda com o intuito de resolver problemas de degradação em redes profundas [43]. Estes problemas são causados pelo uso excessivo de camadas, pois a qualidade das características obtidas vai deteriorando, piorando o desempenho da rede. Desta forma, o ResNet101 foi projetado para reduzir este problema. Introduzindo atalhos, como blocos residuais, é possível que a entrada de um bloco seja adicionada à sua saída, fazendo com que a informação seja mantida ao longo das camadas. Cada bloco residual realiza a equação:

$$y = F(x) + x, \tag{3.16}$$

Onde:

- $F(x)$ representa as operações de convolução e normalização;
- $x$ é a entrada que passa pelo atalho.

Para a diarização de locutor, os vetores extraídos são agrupados por algoritmos similares aos utilizados para agrupar os *i-vectors*, como o K-means. Outro algoritmo que pode ser utilizado é o Variational Bayes HMM Clustering [44].

## 3.4 Modelos

Com as características obtidas, é necessário passar por um modelo que as analise e que permita a obtenção dos resultados pretendidos. Alguns destes são apenas usados para a análise USV, outros apenas na diarização de locutor e outros em ambas as áreas.

### 3.4.1 *Gaussian Mixture Models*

Sendo um modelo probabilístico paramétrico [45], o GMM é utilizado para representar dados que são assumidos como originários de uma combinação de um número finito de distribuições Gaussianas. O modelo tem em vista encontrar parâmetros que melhor se ajustam aos dados observados, sendo:

- Média ($\mu$) - representa o centro da distribuição Gaussiana
- Covariância ($\Sigma$) - define a forma e a dispersão da distribuição
- Peso ($\pi$) - indica a contribuição relativa da componente Gaussiana para o modelo geral

Estes três dados paramétricos, são estimados através do algoritmo de *Expectation Maximization* (EM). O EM é um algoritmo estatístico utilizado para modelos

probabilísticos complexos onde as equações não podem ser resolvidas diretamente, encontrando a máxima verosimilhança para estimar os parâmetros. Este processo está dividido em duas etapas: a expectação ($E$) e a maximização ($M$).

Para a expectação é calculada a probabilidade de cada ponto de dados pertencer a cada componente Gaussiana, utilizando a equação:

$$p_{k,k}(t+1) = \frac{\pi_k(t)N(\zeta_j; \mu_k(t), \Sigma_k(t))}{\sum_{i=1}^{K} \pi_i(t)N(\zeta_j; \mu_i(t), \Sigma_i(t))} \tag{3.17}$$

Onde:

- $\pi_k(t)$ é peso da $k$-ésima componente Gaussiana no tempo $t$;
- $N(\zeta_j; \mu_k(t), \Sigma_k(t))$ é função densidade de probabilidade da distribuição Gaussiana, onde $\mu_k(t)$ é a média e $\Sigma_k(t)$ é a covariância, avaliada no ponto $\zeta_j$;
- $\sum_{i=1}^{K}$ é o somatório que representa a normalização sobre todas as $K$ componentes, garantindo que a soma das responsabilidades seja 1.

Por outro lado, para a maximização, são atualizados os parâmetros para cada componente Gaussiana com base nas probabilidades calculadas, sendo atualizados com as equações:

$$\pi_k(t+1) = \frac{1}{N}\sum_{i=1}^{N} p_{k,j}(t+1) \tag{3.18}$$

$$\mu_k(t+1) = \frac{\sum_{i=1}^{N} p_{k,j}(t+1)\xi_k}{\sum_{i=1}^{N} p_{k,j}(t+1)} \tag{3.19}$$

$$\Sigma_k(t+1) = \frac{\sum_{i=1}^{N} p_{k,j}(t+1)(\xi_j - \mu_k(t+1))(\zeta_j - \mu_k(t+1))^T}{\sum_{i=1}^{N} p_{k,j}(t+1)} \tag{3.20}$$

Onde:

- $p_{k,j}(t+1)$ representa a responsabilidade que o componente $k$ tem sobre a observação $j$ no tempo $t+1$;
- $N$ é o número total de observações;
- $\pi_k$ é a nova probabilidade *a priori* do componente $k$;
- $\mu_k$ é o ajuste da média do componente $k$ ponderando-se cada observação $\xi_k$;
- $\Sigma_k$ é o ajuste da matriz de covariância do componente $k$ ponderando-se cada observação $\xi_j$.

Esta repetição do EM termina após o incremento da verosimilhança logarítmica ser inferior a $e$.

$$L = \sum_{j=1}^{N} log_{10}(p(\zeta_j|0)) \tag{3.21}$$

Onde:

- $L$ representa a soma da verosimilhança logarítmica das observações $\zeta_j$ dadas uma hipótese ou um estado inicial 0.

$$\frac{L_{t+1}}{L_t} < e \tag{3.22}$$

Onde:

- $e$ é um parâmetro de tolerância que define o nível de precisão desejado para o ajuste do modelo;
- $L$ a soma da verosimilhança logarítmica.

O GMM é eficaz na modelagem de dados com múltiplos grupos ou classes, podendo ser utilizado para agrupar diferentes *clusters* com base na sua similaridade e, após ser treinado, poder, também, ser usado para classificar novos pontos de dados em um dos *clusters* existentes.

### 3.4.2 K-means

A utilização de um algoritmo como o K-means permite o *clustering* não supervisionado com o objetivo de separar os dados em K *clusters* distintos. Neste algoritmo, o objetivo é minimizar a distância entre os pontos de dados dentro de cada *cluster*, maximizando a distância entre *clusters* [46].

O primeiro passo a ser realizado é a escolha de pontos de dados para definir as centroides iniciais dos clusters. Esta escolha é definida pela quantidade K de *clusters* pretendida e estes pontos são aleatórios. Em seguida, os restantes pontos são atribuídos aos *clusters* cuja centroide está mais próxima com base numa métrica de dissimilaridade de cosseno. A partir deste passo calcula novas centroides com a equação:

$$c = \frac{1}{N_c} \sum_{j=1}^{N_c} x_j \tag{3.23}$$

Onde:

- $N_c$ é o número de pontos no *cluster*;
- $x_j$ é um determinado ponto.

Assim, constrói novas partições, novamente, baseadas nas distâncias a que os pontos estão das centroides.

O cálculo da distância, como mencionado, é através da dissimilaridade de cosseno, que é calculado da seguinte forma [47]:

$$Cs|x_{.}y| = \frac{xy}{\parallel x \parallel \times \parallel y \parallel} \tag{3.24}$$

$$Cd|x_{.}y| = 1 - Cs \tag{3.25}$$

Onde:

- $Cs$ é o resultado da similaridade de cosseno;
- $Cd$ é o resultado da dissimilaridade de cosseno;
- $xy$ é o produto interno dos vetores x e y;
- $\parallel x \parallel$ e $\parallel y \parallel$ são as normas dos vetores x e y.

O K-means, na prática, pode ser aplicado para agrupar *embeddings* de locutor em *clusters*, como também pode ser aplicado na análise de dados, agrupando características com base em padrões.

### 3.4.3 *Hidden Markov Models*

Os *Hidden Markov Models* (HMM), como os GMM, são modelos estatísticos. Estes modelos, utilizam uma sequência de observações como sendo geradas a partir de uma outra sequência de estados ocultos subjacentes. Como para os modelos previamente abordados, o HMM utiliza um valor pré definido de estados ocultos na qual, como o método de GMM, calcula a probabilidade conjunta das observações e dos estados ocultos [48]. Características como os MFCC podem ser utilizadas como dados num HMM.

O processo de treino de um HMM envolve encontrar os valores que maximizam a probabilidade dos dados observados. Esta otimização pode ser facilitada utilizando um algoritmo como o de Baum-Welch, um tipo de algoritmo EM. Este tipo de algoritmo permite treinar um modelo HMM para reconhecer padrões em sequências de dados. Como referido na Subsecção 3.4.1, um algoritmo EM é dividido em duas etapas, uma primeira de expectação ($E$) e uma segunda de maximização ($M$). O mesmo se aplica no algoritmo de Baum-Welch.

Os HMM têm, no entanto, algumas limitações. Uma das mais relevantes é a criação de um número elevado de parâmetros não estruturados quando se tem muitos estados ocultos. Isto pode levar a uma inconsistência durante a fase de treino como na interpretação de resultados [49].

## 3.5 Resumo

As explicações dos vários métodos utilizados nos projetos foram abordadas ao longo do Capítulo 3. Numa primeira instância, foi realizada uma breve explicação do que são as *Ultrasonic Vocalizations*, porque são estudadas e alguns dos significados que possam ter.

Por outro lado, na Secção 3.2 foi feita uma análise detalhada de como são obtidas as características acústicas de um sinal, tendo em consideração o tema do projeto. Primeiro, foram explorados os sonogramas, dado que são utilizados numa grande maioria dos projetos de análise de USV. Ao longo da sua explicação, foram abordados os vários passos que são tomados, desde a segmentação do sinal até ao cálculo da magnitude com base no resultado da Transformada de Fourier. A segunda característica abordada foi a dos MFCC, onde alguns dos passos iniciais foram omitidos, devido a serem os mesmos dos sonogramas. Por fim, foi analisado as características prosódicas, visto serem utilizadas para complementar as características dos MFCC. Estas variam de projeto para projeto, dependendo do objetivo. No entanto, as principais características foram abordadas de forma a esclarecer a sua obtenção.

Na Secção 3.3 foi realizado o estudo dos *embeddings*, as características utilizadas em modelos de *machine learning* e *deep learning*. Aqui foram abordados as duas mais utilizadas, *i-vectors* e *x-vectors*. O mesmo processo feito para as características acústicas é feito para estes dois, foi realizado uma análise de como são obtidos estes dados e como são utilizados.

Por fim, foi composto uma análise de alguns modelos utilizados, o *Gaussian Mixture Models*, o K-means e o *Hidden Markov Models*. Os modelos apresentam resultados semelhantes, onde ambos fazem *clustering* para um número pré definido de *clusters*, mas as técnicas utilizadas para a obtenção de ambos são distintas.

# Capítulo 4

# Implementação

O desenvolvimento do projeto passou por várias fases, os vários (in)sucessos das ideias que conduziram ao resultado final. Assim, é imprescindível explicar a evolução do projeto, pois os métodos inicias podiam ser válidos noutras situações e foram estes que conduziram ao final. Neste capítulo, são abordadas várias fases, de forma cronológica, começando por simples testes de visualização dos sinais até à obtenção de um sinal capaz de identificar a comunicação.

## 4.1 Requisitos

Para a resolução do "problema"é necessário a criação de um protótipo com as seguintes características:

- Ser capaz de ler ficheiros .WAV;
- Analisar esse ficheiro de maneira a conseguir distinguir a progenitora da cria;
- Mostrar gráfico, o espectro do ficheiro juntamente com o resultado obtido da análise;
- Ter uma interface de fácil utilização.

### 4.1.1 Diagrama de blocos

Para a criação do projeto, foi desenhado o diagrama de blocos, presente na Figura 4.1, onde é possível observar as quatro funcionalidades do protótipo.

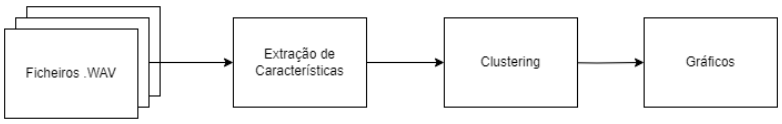



Figura 4.1: Diagrama de Blocos

1. É feita a escolha e leitura de um ficheiro .WAV, onde o utilizador deverá escolher o ficheiro;
2. É realizada a extração de características;
3. É feito um *clustering* com base nessas características;
4. É mostrado um gráfico com os dados obtidos nos passos anteriores, onde o utilizador poderá observar momentos específicos.

## 4.2 Desenvolvimento

O início do protótipo não começou com a construção de um modelo seguindo o diagrama da Figura 4.1. O primeiro passo realizado foi a criação de um simples sistema capaz de observar segmentos do espectro e só posteriormente foi iniciada a criação do protótipo.

### 4.2.1 Preparação

Através do Jupyter Notebook, foi realizado um programa para observar um segmento do espectro. Todo o desenvolvimento foi feito com o intuito de ser algo temporário, com a consciência de que o que seria criado teria pouca aplicabilidade para o programa final. No entanto, esta primeira fase foi essencial para uma melhor compreensão de como é uma comunicação entre progenitora e cria de murganhos. Para a obtenção dos resultados, desenvolveu-se um código onde, manualmente, foi definido o ficheiro e o intervalo de tempo para ser observado. Neste, através das bibliotecas scipy [50], para a leitura e tratamento do sinal, e Matplotlib [51], para a criação dos gráficos, é visualizado um pequeno aglomerado de vocalizações.

Na Figura 4.2,podem observar-se alguns dos vários sinais visualizados. É importante salientar que estes valores foram escolhidos após observar-se possíveis períodos onde, num ficheiro de vídeo, os murganhos estavam próximos do microfone. Este ficheiro foi útil para ajudar a encontrar comunicações das crias, da progenitora ou de ambas. Continuando a análise dos gráficos, é notório que as frequências de comunicação da progenitora são inferiores às das crias, podendo ver-se uma separação perto dos 60000 Hz. Apesar de não ser um resultado exato, estes dados, serão úteis para uma melhor compreensão dos resultados finais. Por outro lado, sempre que a

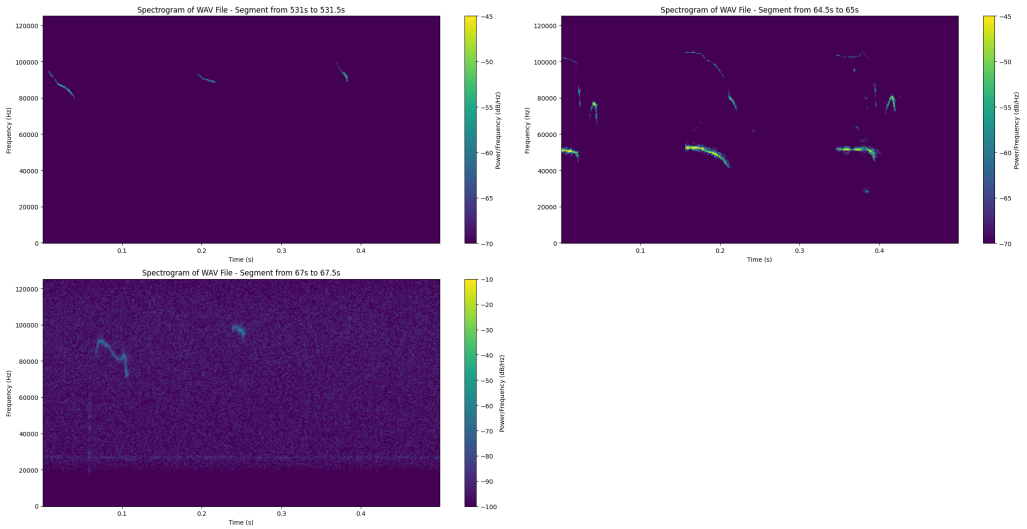



Figura 4.2: Três gráficos com o espectrograma

progenitora comunica, observa-se uma repetição da sua comunicação com o dobro da frequência. Este fenómeno é um harmónico, não sendo uma repetição feita pela cria [52]. Além disso, de forma a obter-se a qualidade observada, as variáveis de nperseg e de noverlap foram definidas como 512 e 256, respetivamente.

### 4.2.2 Exploração de Técnicas de *Clustering*

Após compreender-se como são as comunicações, o objetivo seguinte foi a criação do protótipo capaz de analisar autonomamente o sinal. Para isso, teve de se fazer a escolha do método de clustering seria utilizado e como seria implementado. São mais indicados os métodos de *clustering*, onde apenas se fornecem os dados do sinal, não necessitando de características adicionais para definir parâmetros. Assim, optou-se por testar vários métodos de forma a escolher-se o que teria o resultado mais indicado. Além disso, bibliotecas como sklearn [53] contêm vários algoritmos já criados. Dos vários métodos disponíveis, escolheu-se o método de K-means e *Gaussian Mixture Models*, previamente estudados, mas também se optou por experimentar outros métodos como Agglomerative Clustering, Spectral Clustering e Birtch para se obter uma comparação.

## 4.3 Versão Final

O código tinha de ter certas características como a existência de uma interface para uma fácil interação, sem a necessidade de interagir com o código para trocar de ficheiros, escolher funcionalidades ou, até mesmo, definir um período de visualização. Para tal, o Jupyter Notebook não era mais prático, passando assim para Python. Para a execução do novo código, passou-se a utilizar bibliotecas sklearn, librosa [54],


Figura 4.3: Seleção de ficheiro

IPython[55], ipywidgets e TKinter, para além das duas previamente referidas no capítulo 4.1. A biblioteca librosa, permite a leitura do ficheiro áudio sem aparecer uma mensagem a informar que tinha ignorado um pedaço de *non-data*, que aparecia ao usar o scipy. Outra funcionalidade do librosa é a possibilidade de obter *Mel Frequency Cepstral Coefficients* (MFCC), tendo sido esta a *feature* escolhida. As bibliotecas IPython, ipywidgets e TKinter foram escolhidas para fazer a interface gráfica.

### 4.3.1 Extração de Características

A escolha dos ficheiros é feita através de um botão, que ao ser premido abre um *pop up*, onde o utilizador deve navegar até à pasta e selecionar o ficheiro.

Na Figura 4.3 pode observar-se o botão e o aspeto do *pop up* com a localização da pasta, neste caso, apresenta uma pasta com alguns ficheiros de áudio.

O segundo passo acontece após premir-se o segundo botão, onde é feita uma filtragem de todas as frequências inferiores a 20 kHz, visto que os murganhos comunicam a frequências superiores. Para além disso, são obtidas as *features* por MFCC. Os parâmetros definidos foram obtidos com base no melhor resultado obtido em testes e são:

- Número de MFCC - 40
- Número de FFT - 4096
- *Hop lenght* - 2048

```
MemoryError: Unable to allocate 86.1 GiB for an array with shape (107502, 107502) and data type float64
```

(a) Mensagem de erro para o método de Spectral Clustering

```
MemoryError: Unable to allocate 43.1 GiB for an array with shape (5778286251,) and data type float64
```

(b) Mensagem de erro para os métodos de Agglomerative Clustering e Birtch

Figura 4.4: Mensagem de Erro de memória RAM

### 4.3.2 Clustering

Ao contrário dos dois passos anteriores, que são executados em botões diferentes, *clustering* é uma continuação da obtenção das características, ou seja, logo a seguir a extrair as *features* do áudio, é feito o *clustering* tanto para o método de K-means como para o método de GMM. É definido o número de *clusters* como 4, para cada representar as 4 situações em questão, comunicação da progenitora, comunicação da cria, comunicação de ambos ou ruído. É importante realçar que os valores de cada *cluster* não têm o mesmo significado para todos os ficheiros, isto porque não existe uma definição para cada um, isto é, a cada processamento, é necessário verificar o que cada *cluster* representa. No capítulo 5 será visível esta distinção.

Para o K-means, foram, ainda, definidos os parâmetros de *random state*, iterações máximas e número de inicializações, 42, 3000 e 100, respetivamente. Estes valores foram valores que funcionaram e o *random state* foi definido para que no mesmo ficheiro, o resultado fosse consistente.

As definições de parâmetros foram realizadas para o método de GMM, onde apenas os parâmetros *random state* e iterações máximas foram definidos, para 42 e 3000, respetivamente. Como no método anterior, o *random state* foi definido para dar consistência aos valores obtidos e o valor de iterações foi o obteve os resultados pretendidos.

#### Problemas no *clustering*

Ao longo do desenvolvimento, foram notados alguns problemas que levaram a alterações do plano inicial. Alguns métodos de *clustering* como os Agglomerative Clustering, Spectral Clustering e Birtch utilizavam demasiada memória RAM, em certas instâncias, a consola refere que não consegue realizar o processamento do áudio, por não conseguir alocar toda a informaçãoque, dependendo do método, podia exceder os 40 GB. É importante realçar que o computador utilizado, apenas possui 16 GB de memória RAM. Assim os métodos não podem ser uma opção.

Na Figura 4.4, pode observar-se o problema de alocação de memória para os vários métodos, onde o método de Spectral Clustering precisa de alocar 86,1 GiB e os métodos de Agglomerative Clustering e Birtch necessitam de 43,1 GiB. Estes dois valores em GB são 92,4 e 46,3, respetivamente.

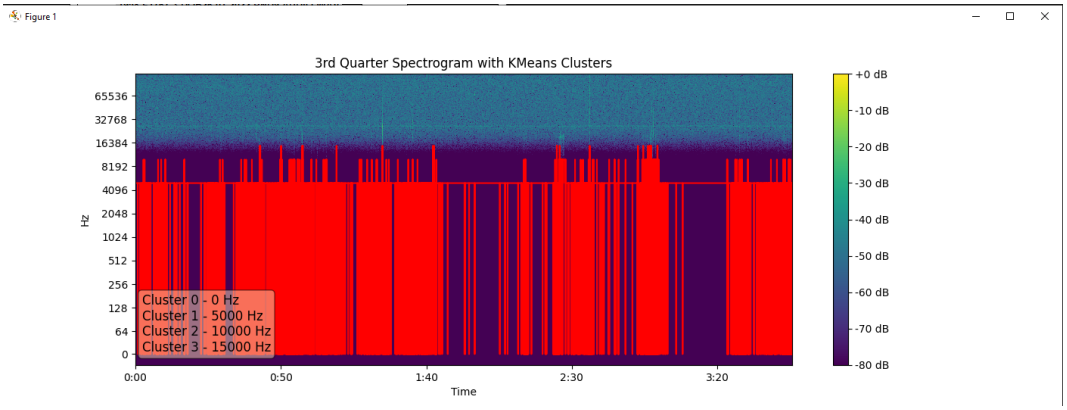



Figura 4.5: Exemplo do resultado do gráfico com *clustering* através de K-means

Dos vários métodos testados, apenas o método de K-means e de GMM são os que conseguem funcionar nas presentes circunstâncias.

### 4.3.3 Gráficos

Para a criação dos gráficos, o processo ficou mais complexo que o esperado devido à utilização excessiva de memória. Como será referido, o gráfico não pode ser representado na sua totalidade, passando a ser visualizado em quatro partes.

Na interface gráfica foram definidos quatro botões, um para cada quarto do sinal. No botão do primeiro quarto, é calculado o espectro apenas do período correspondente. Com esses valores, são realizados um total de dois gráficos sequenciais, um para cada método de *clustering*, onde o primeiro é o Gaussian Mixture e o segundo o K-means. Este processo repete-se nos para os outros três botões, mas com o período que corresponde ao botão premido.

Em cada gráfico existe uma legenda contendo o significado dos valores do *cluster*. Como os valores variam entre 0 e 3 e o gráfico apresenta-se numa escala que varia entre os 0 Hz e os 120000 Hz, os valores tiveram de ser multiplicados para poderem ser visualizados em conjunto com o espectro. Assim, o *cluster* 0 corresponde ao valor 0, o *cluster* 1 ao valor 5000, o 2 ao 10000 e, por fim, o 3 corresponde ao valor 15000.

Relativamente ao espectro, os valores utilizados para as variáveis n_fft e hop_length foram 4096 e 512, respetivamente, pois foram os valores encontrados que permitiram ter uma visualização clara do espectro.

Na Figura 4.5, pode se observar um exemplo de um dos gráficos obtido. Neste identifica-se que representa o terceiro quarto do espectro do áudio e que os valores de *cluster* obtidos são do método de K-means.

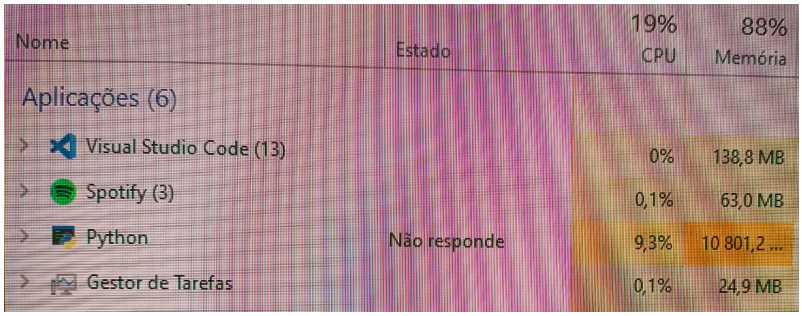


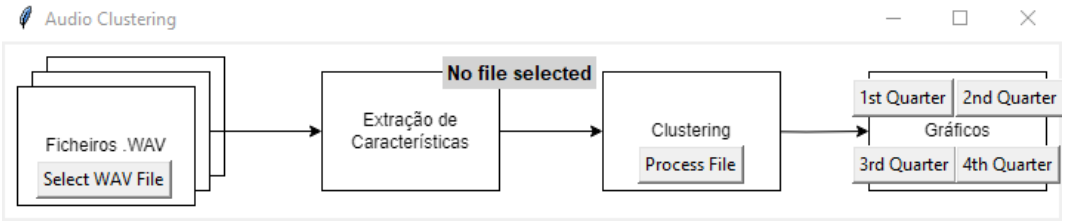

Figura 4.6: Utilização de RAM ao correr o programa (a qualidade da imagem é reduzida devido a ser uma fotografia, porque não era possível fazer um *print screen*)

### Problemas nos gráficos

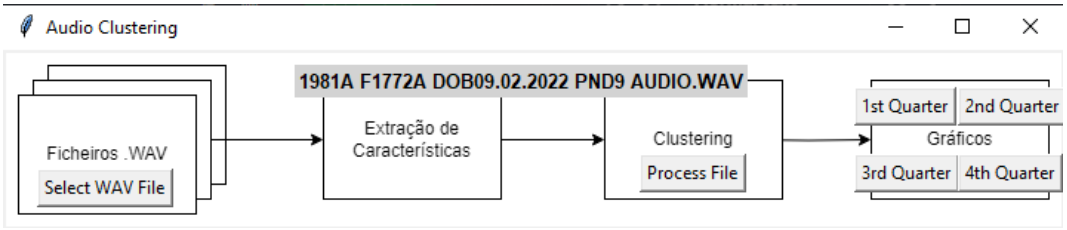
Para a criação dos gráficos, o processo ficou mais complicado do que o esperado, devido à utilização excessiva de memória RAM ao observar o espectograma. Inicialmente, quando se tentava fazer o *plot* do gráfico, o computador não conseguia fazer o gráfico, o que levava o Python a não responder, sem aparecer o gráfico, e outros programas paravam de funcionar ou até mesmo fechavam. Quando se percebeu que o problema era causado pela insuficiência de memória RAM, o ficheiro passou a ser mostrado em duas metades, com uma opção para cada metade na interface. No entanto, devido ao facto de serem muitos dados para visualizar, a análise de um período de comunicações teve de ser realizada com a ampliação da específica região, o que levava a um novo *plot*. Este processo, de *zoom in* e *zoom out*, era muito demorado e, na observação de um segundo gráfico, era necessária a utilização de mais memória RAM. Desta forma, cada metade voltou a ser dividida ao meio, passando a ser quatro botões. Assim, os valores de memória e o tempo de processamento foram reduzidos. No entanto, mesmo com uma sucessiva visualização dos gráficos, o mesmo problema voltava a surgir. Para garantir cada leitura, definiu-se que após visualizar-se os gráficos, os dados do espectro seriam apagados da memória. Deste modo, ficou possível ver e rever os gráficos sem que o computador ficasse sem memória. Através de leituras manuais do valor da memória, no gestor de tarefas, esta alteração não é observável, pois estes continuam a atingir valores similares, perto dos 11 GB, mas os gráficos foram sempre apresentados, nos vários testes realizados.

Na Figura 4.6, é possível observar a utilização de memória que acontece ao executar certas funcionalidades. Esta imagem obteve-se através de uma fotografia, visto não ser possível obtê-la pelo computador. Por outro lado, é impossível mostrar o gráfico não abrir, como também é impossível mostrar os outros programas não funcionarem

Com base nos problemas referidos, algumas alterações tiveram de ser implementadas: a criação de quatro botões para se observar os gráficos e apagar os dados da memória.




(a) *Pop up* no início




(b) *Pop up* após a escolha do ficheiro

Figura 4.7: Evolução do *pop up* ao longo da utilização do programa

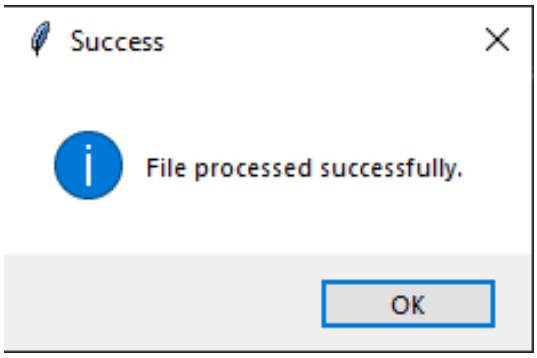


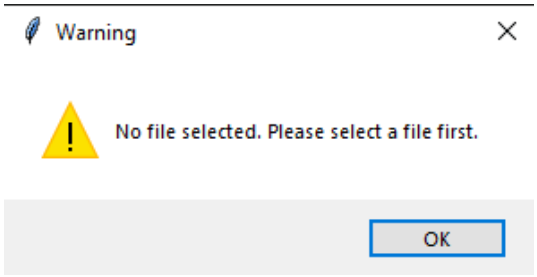

Figura 4.8: Mensagem a confirmar que o ficheiro foi processado

## 4.4 Funcionamento

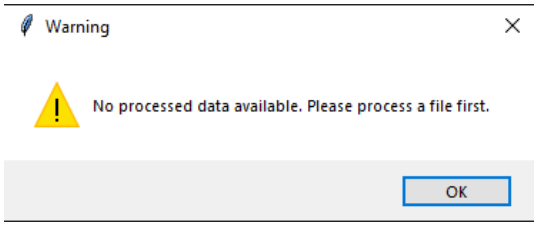
Com o programa a correr, o utilizador encontra primeiro o diagrama de blocos como fundo de um *pop up* com botões. Na Figura 4.7a, é possível observar esse *pop up* ca informação extra que não tem qualquer ficheiro selecionado. Como visível na Figura 4.7b, após a escolha do ficheiro, a área de texto passa a conter o nome desse mesmo ficheiro.

Depois da escolha do ficheiro, deverá ser premido o botão sobre o bloco de *clustering*, no qual, após o processamento, uma mensagem irá informar que está concluído. Essa imagem pode ser observada na Figura 4.8. Após fechar a mensagem, o utilizador deverá premir o botão para um dos quartos do gráfico no qual, dois gráficos vão aparecer de forma sequencial. Na Figura 4.5 tem-se o que será previsto ser o resultado de um dos gráficos.

É importante referir que para fazer o gráfico é necessário fazer, primeiro, o processamento e antes desse processamento é preciso selecionar um ficheiro. Desta




(a) Mensagem de erro devido a não estar nenhum ficheiro selecionado




(b) Mensagem de erro devido a não estar processado

Figura 4.9: Mensagens de erro para caso o utilizador tenha avançado passos

forma, uma mensagem de erro irá aparecer, a informar que é necessário selecionar um ficheiro ou processá-lo, como aparece na Figura 4.9. No entanto, estas mensagens não irão aparecer na situação de se trocar de ficheiro, ou seja, após processar-se um primeiro ficheiro, ao trocar para um segundo, nenhuma mensagem de aviso irá aparecer caso não se tenha feito o processamento, mostrando assim o resultado do primeiro áudio.

## 4.5 Resumo

Ao longo do Capítulo 4, foi explorado todo o processo de desenvolvimento do projeto. Começou-se por definir os seus requisitos do projeto com a ajuda de um diagrama de blocos, que continha a informação necessária. Na Secção 4.2, foi realizada uma descrição dos vários passos tomados, desde uma fase primária, onde o foco não era o desenvolvimento de um programa para identificar os murganhos, mas sim, de um programa que permitisse a observação das vocalizações. Quando se começou a planear os primeiros desenvolvimentos do programa final, foram exploradas algumas técnicas de *clustering*.

A versão final do projeto foi explorada ao longo da Secção 4.3, onde se examinaram as bibliotecas utilizadas e as alterações realizadas, face à projeção inicial. Esta secção abordou os principais tópicos do código, desde a extração de características até aos métodos de *clustering* escolhidos. Ao longo da explicação foram salientados alguns dos problemas passados que explicam o porquê de certas decisões, como por

exemplo: a alteração de mostrar o resultado em quatros gráficos diferentes, em vez de mostrar os resultados num só gráfico.

Por fim, na Secção 4.4, foi realizada uma breve explicação de como utilizar o programa, analisando a ordem das ações a serem tomadas, e os efeitos de não se seguir essa sequência.

# Capítulo 5

# Resultados e Comentários

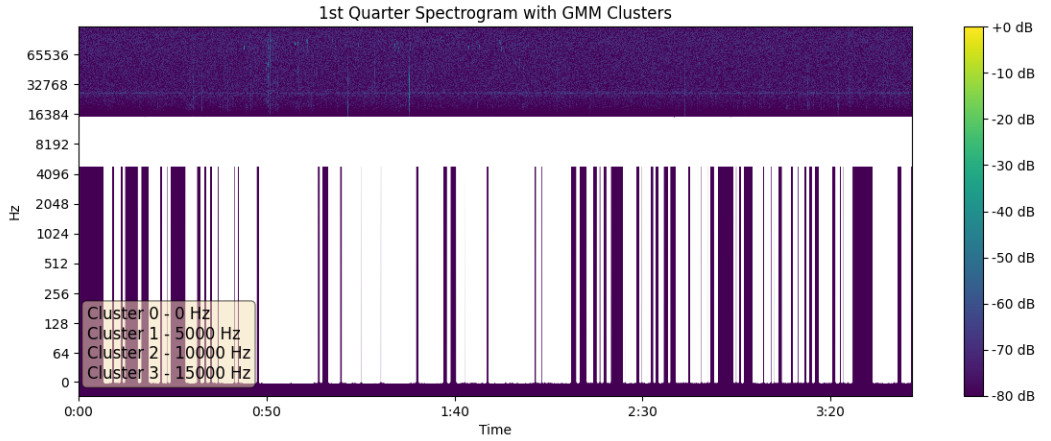
Com o desenvolvimento do programa concluído, pode-se, por fim, verificar e avaliar os resultados obtidos. Neste capítulo, é, assim, efetuada uma avaliação dos resultados, comparando diferentes situações, para se verificar se, de facto, se conseguiu responder ao problema: ser capaz de distinguir as vocalizações entre a progenitora e as suas crias, de forma que seja possível separar e distinguir as vocalizações.

## 5.1 Resultados obtidos

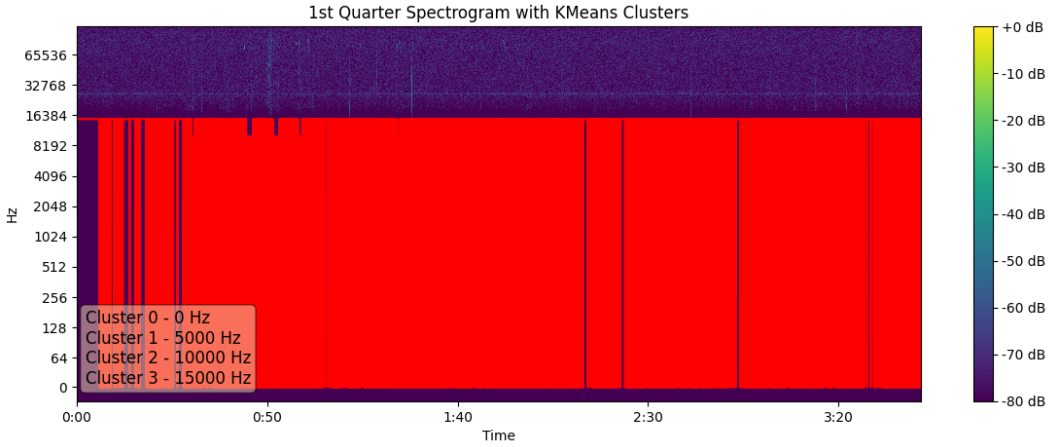
Seguindo os passos de funcionamento, obtém-se um resultado como o da Figura 5.1, onde, inicialmente, o gráfico aparece com os dados obtidos pelo método de Gaussian Mixture, Figura 5.1a, e, posteriormente, pelo método de K-means, Figura 5.1b.

Para se poder analisar os resultados dos *clusterings*, é necessário observar várias comunicações, presentes nos áudios. Para tal, vai-se explorar alguns resultados, comparando, primeiro, os valores individualmente e, posteriormente, entre todos. É importante salientar que os excertos apresentados, são apenas exemplos onde existe uma maior discrepância no tipo de vocalizações presentes.

Nas Figuras 5.2 e 5.3 é possível observar-se, no mesmo áudio, dois segundos distintos, onde nas Figuras 5.2a e 5.2b está representado o segundo 50 do sinal com o resultado obtido por Gaussian Mixture e por K-means. Nas Figuras 5.3a e 5.3b o mesmo está representado para o segundo 58. Observando os espectros de cada excerto, é possível observar a disparidade nos espectros, realçando, assim, a variedade de vocalizações que os murganhos fazem.

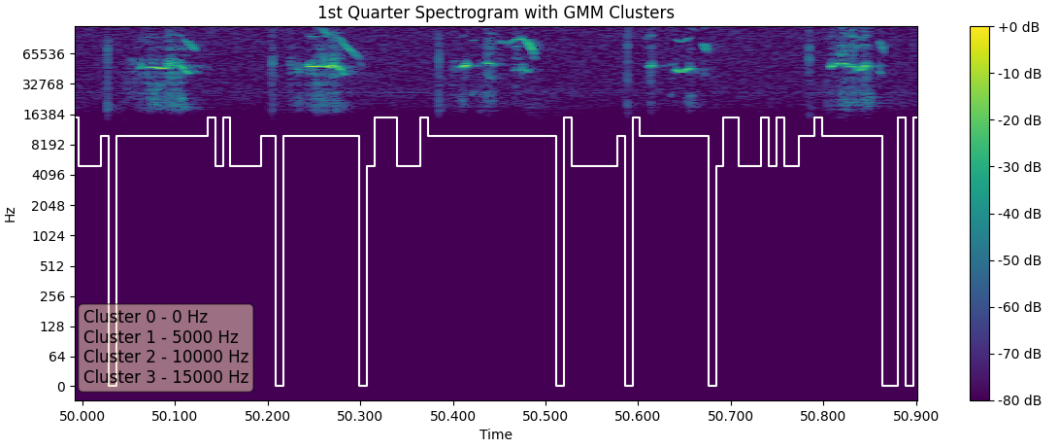



(a) Resultado obtido para Gaussian Mixture

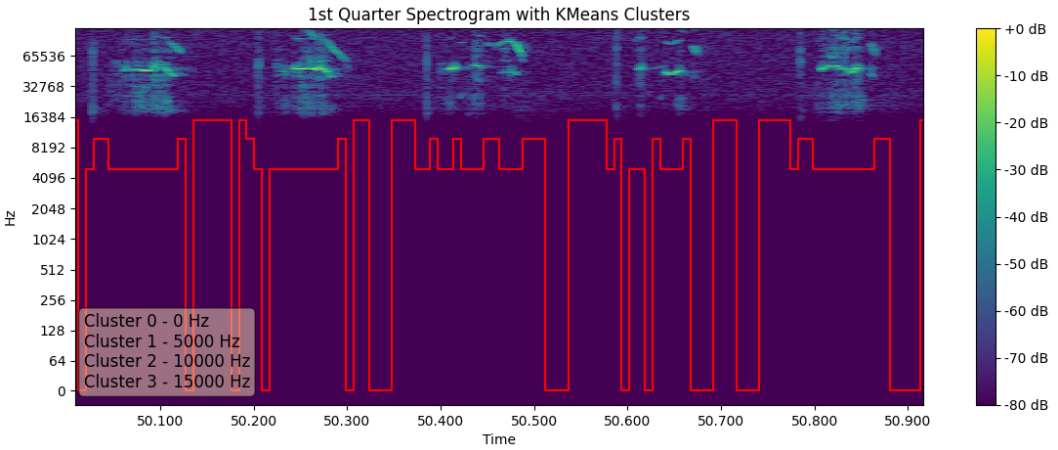



(b) Resultado obtido para K-means

Figura 5.1: Resultado exemplo para quando se carrega no botão para apresentar os gráficos

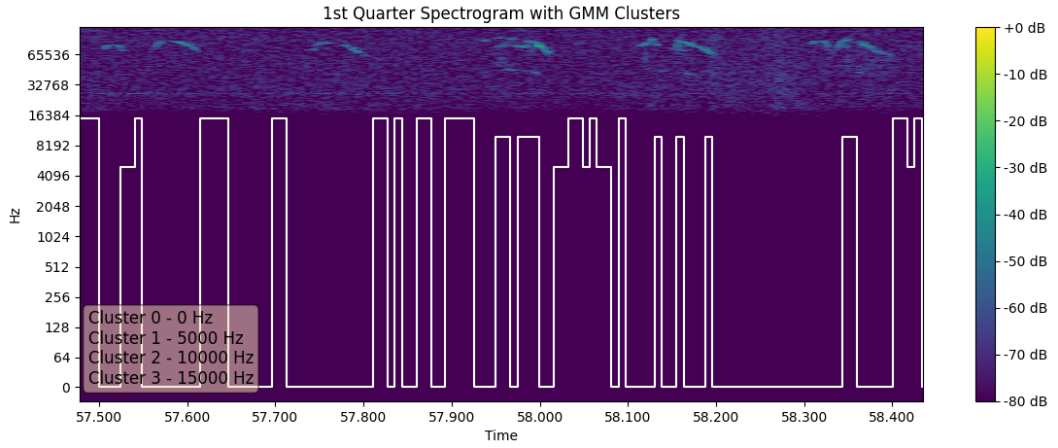



(a) Resultado obtido para Gaussian Mixture do segundo 50

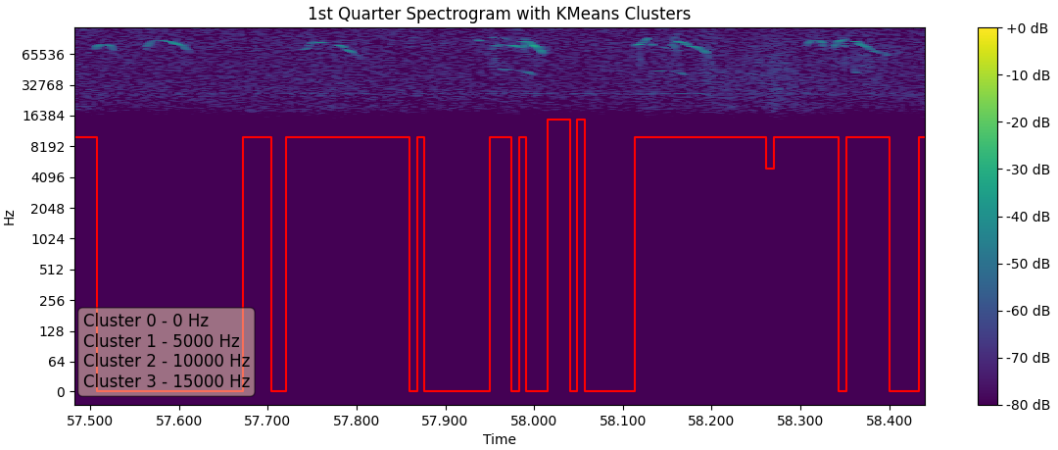



(b) Resultado obtido para K-means do segundo 50

Figura 5.2: Resultado obtido no segundo 50 do áudio




(a) Resultado obtido para Gaussian Mixture do segundo 58




(b) Resultado obtido para K-means do segundo 58

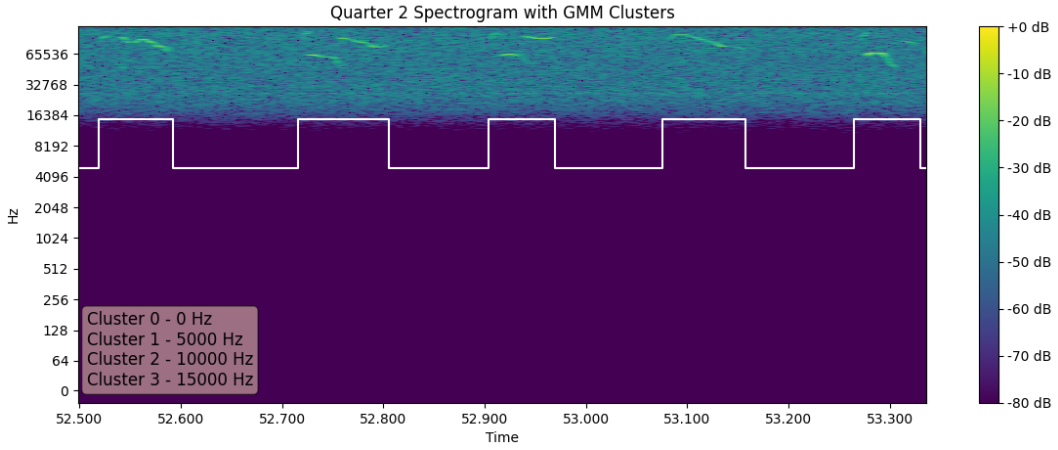
Figura 5.3: Resultado obtido no segundo 58 do áudio

Começando pelo segundo 50, na Figura 5.2, é possível observar que, no método de Gaussian Mixture, quando o espectro tem uma maior intensidade, perto dos 50000 Hz, o valor dos *clusters* encontra-se representado na faixa dos 10000 Hz, ou seja, no *cluster* número 2. No entanto, os resultados obtidos noutras partes do espectro não aparentam ser consistentes, variando noutros valores. Por outro lado, nos resultados para K-means, os valores dos *clusters* são mais variados ao longo do espectro. Na vocalização entre 50,200 e 50,300 segundos, o *cluster* tem o valor equivalente a 1, quando a maior amplitude se encontra nos 50000 Hz. O valor varia para 2 quando a uma frequência superior começa a existir uma amplitude considerável. Quando a intensidade nas frequências de 50000 Hz diminui, o valor do *cluster* altera para 0 até a vocalização, acima dos 65000 Hz, terminar. Um resultado similar pode ser observado nas vocalizações entre 50,400 e 50,500 s e 50,600 e 50,700 s.

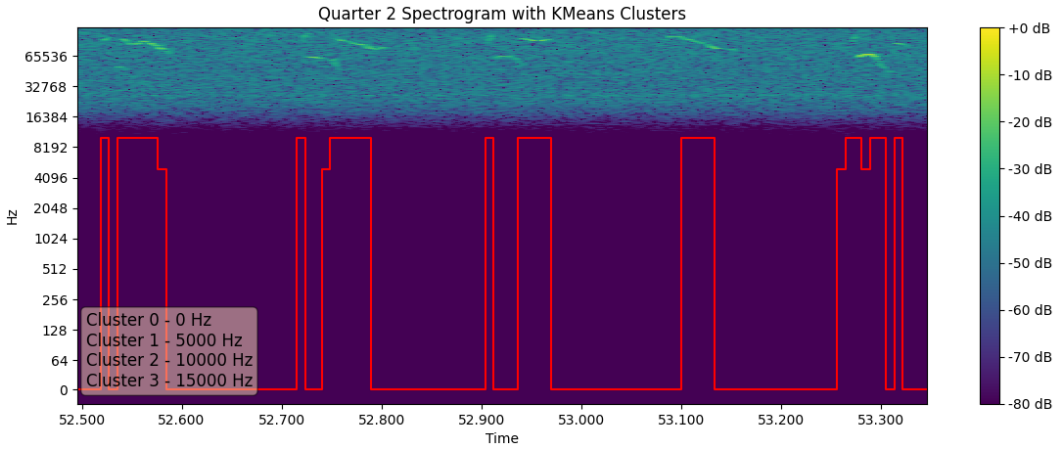
No segundo 58, as vocalizações têm frequências diferentes. Nestas, ao contrário das anteriores, todas as vocalizações encontram-se acima dos 65000 Hz. Na Figura 5.3a, é possível observar os valores dos *clusters* para o método de Gaussian Mixture. Do mesmo modo que nos resultados anteriores, no método de Gaussian Mixture, os valores são consistentes durante as vocalizações. Neste caso tomando o valor do *cluster* 0, e nas situações em que também existem vocalizações nas frequências abaixo de 65000 Hz, o valor do *cluster* altera para 2. Para o método de K-means, presente na Figura 5.3b, ao longo das vocalizações, superiores a 65000 Hz, o *cluster* definido é o 0 e nas vocalizações em frequências tanto acima como abaixo dos 65000 Hz o *cluster* definido toma o valor 2. No entanto, este valor também é definido na comunicação entre 57,700 e 57,800 s onde não são visíveis vocalizações abaixo de 65000 Hz.

Nas Figuras 5.4 e 5.5, observa-se um efeito diferente no espectro. Aqui, estão apresentados dois excertos que mostram um espectro com vocalizações mais simples que as apresentadas nas Figuras 5.2 e 5.3.

No segundo 53, encontram-se vocalizações tanto acima da frequência de 65000 Hz, como abaixo ou simultaneamente acima e abaixo. Para o método de Gaussian Mixture. Na Figura 5.4a, os valores dos *clusters* são sempre os mesmos para cada vocalização, *cluster* 3, e quando não é uma vocalização, o valore é 1. É importante salientar que os valores dos *clusters* têm significados diferentes neste áudio. O mesmo acontece para o método de K-means. Na Figura 5.4b, encontra-se o método de K-means. Aqui os valores dos *clusters* não são tão constantes como no método anterior. Por exemplo, na primeira vocalização, presente entre 52,500 e 52,600 s, o *cluster* varia quando existe uma breve vocalização abaixo dos 65000 Hz, passando do *cluster* 2 para o 0. Nas vocalizações com frequências tanto acima como abaixo dos 65000 Hz é mais clara esta distinção, onde nas situações em que apenas têm comunicações inferiores a 65000 Hz o valor do *cluster* é 0, quando tem tanto acima como abaixo, é 1 e, enquanto as comunicações se encontram acima dos 65000 Hz, o valor é 2. No

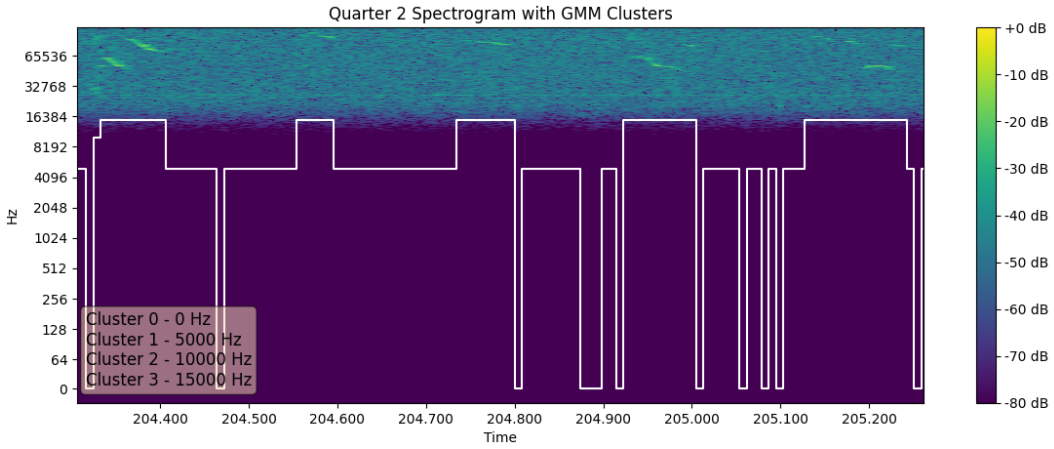



(a) Resultado obtido para Gaussian Mixture do segundo 53 do segundo quarto do áudio

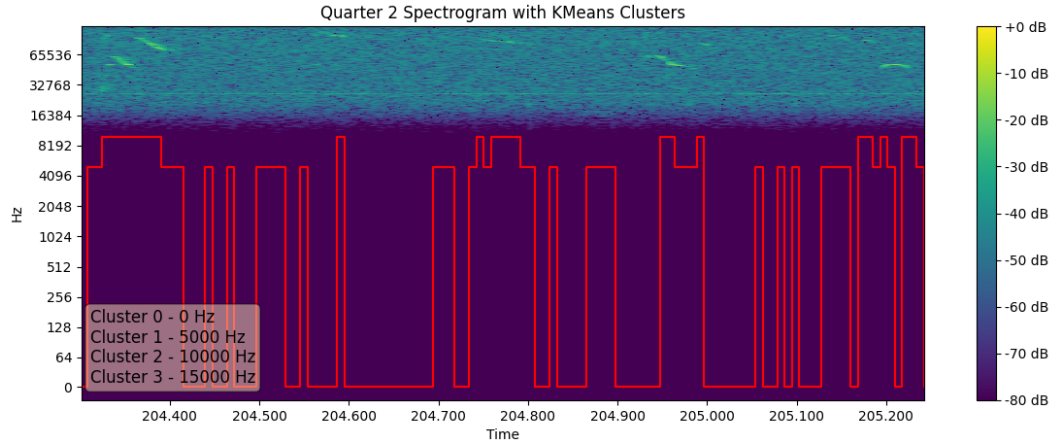



(b) Resultado obtido para K-means do segundo 53 do segundo quarto do áudio

Figura 5.4: Resultado obtido no segundo 53 do segundo quarto do áudio




(a) Resultado obtido para Gaussian Mixture do segundo 204 do segundo quarto do áudio




(b) Resultado obtido para K-means do segundo 204 do segundo quarto do áudio

Figura 5.5: Resultado obtido no segundo 204 do segundo quarto do áudio

entanto, entre vocalizações, o valor definido pelo *cluster* é 0 também.

Para o segundo 204, os valores dos *clusters*, para o método de Gaussian Mixture, são mais variados que em relação ao segundo 53. Na Figura 5.5a, observa-se que durante as vocalizações, independente de qual seja, o valor do *cluster* mantém-se no 4. Desta forma, todas as variações neste método só acontecem fora das comunicações, ou seja, no ruído. Para o método de K-means, na Figura 5.5b, também existe uma grande variação dos valores do *cluster* fora das vocalizações.

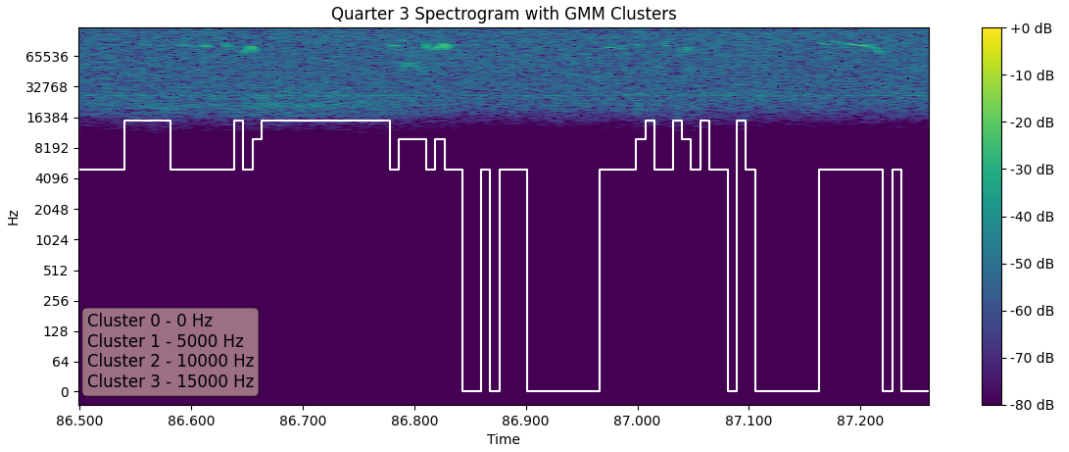
Por fim, escolheu-se os espectros presentes nas Figuras 5.6 e 5.7. Estes dois foram escolhidos devido à complexidade intermédia quando comparados com os quatro exemplos previamente explorados. No método de Gaussian Mixture, do espectro na Figura 5.6a, pode-se ver que durante as vocalizações superiores a 65000 Hz o valor do *cluster* é 1 e toma o valor 2 quando também existe uma vocalização com frequências abaixo de 65000 Hz, como na vocalização que decorre no instante 86,600 s. Por outro lado, no método de K-means, durante as vocalizações superiores a 65000 s, o valor tomado pelo *cluster* é 3 e é 0 quando está em frequências inferiores a 65000 Hz. Em ambos os métodos, entre vocalizações é possível observar várias variações do valor do *cluster*.

Na Figura 5.7, ambos os métodos têm resultados similares, tendo apenas dois valores no *cluster*, um para quando está a haver vocalizações e outro para quando não há vocalizações. Este resultado surge pelo facto de todas as comunicações presentes serem superiores a 65000 Hz. No caso do método de Gaussian Mixture, o valor 1 encontra-se quando há vocalizações e o valor 0 quando não há vocalizações, visível na Figura 5.7a. Para o método de K-means, Figura 5.7b, os valores são de 1 quando é uma vocalização e 2 quando não é.

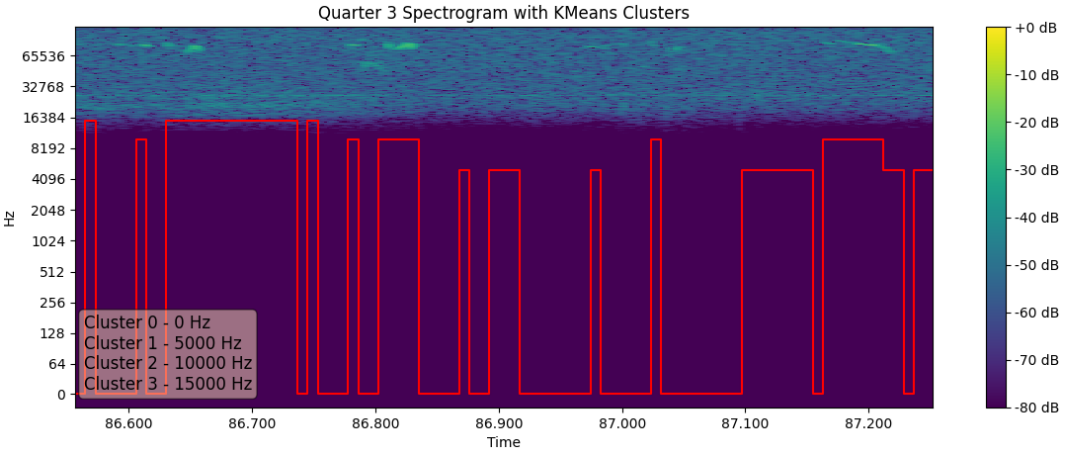
## 5.2 Resumo

Ao fazer uma análise do Capítulo 5, foi possível concluir que utilizando métodos de *clustering*, como Gaussian Mixture e K-means, as vocalizações foram detetadas, em todos os resultados. Neste caso, grande parte das vocalizações pertenciam a um *cluster* bem definido.

No entanto, o resultado obtido entre vocalizações foi, na maior parte das vezes, sensível. Gráficos como os das Figura 5.3a, Figura 5.4a e Figura 5.7b são um bom exemplo do contraste existente. Nos três detetaram-se corretamente as vocalizações, mas apenas no segundo e terceiro existiu uma leitura constante das zonas onde não estão presentes comunicações. No entanto, os métodos não tiveram resultados tão semelhantes quando se fala de identificação de vocalizações. O método de K-means teve uma maior variação durante as vocalizações, trocando de estado mais frequentemente quando uma alteração acontecia ao espectro, por exemplo nas Figuras 5.2b e 5.5b, é possível ver que este método foi mais suscetível a pequenas variações, sendo

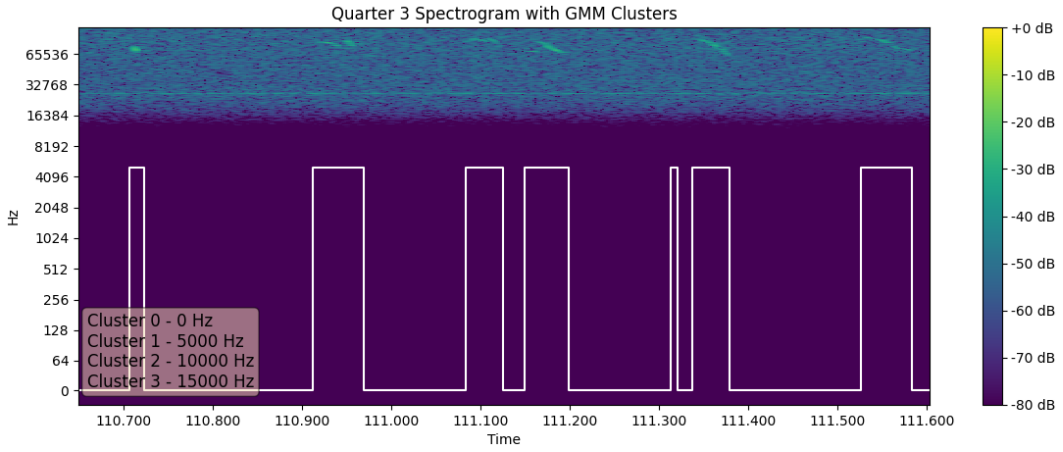



(a) Resultado obtido para Gaussian Mixture do segundo 87 do terceiro quarto do áudio

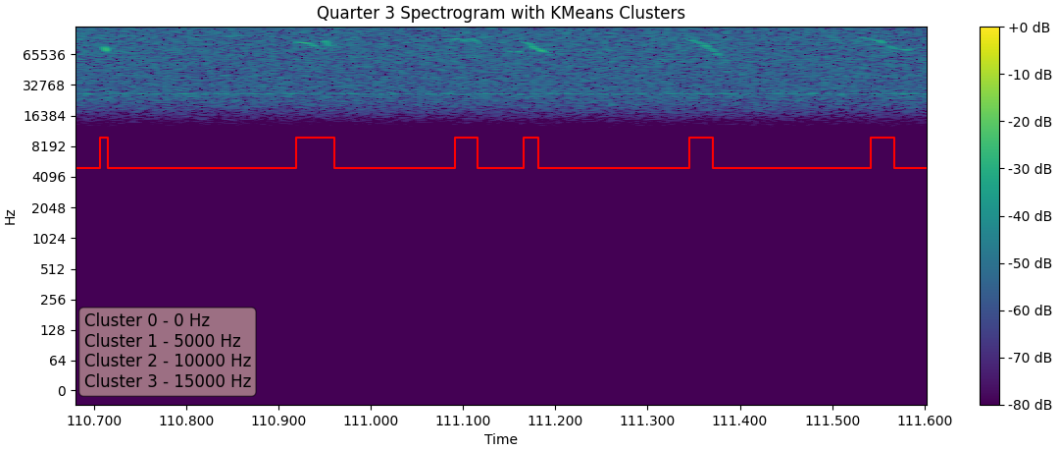



(b) Resultado obtido para K-means do segundo 87 do terceiro quarto do áudio

Figura 5.6: Resultado obtido no segundo 87 do terceiro quarto do áudio




(a) Resultado obtido para Gaussian Mixture do segundo 111 do segundo quarto do áudio




(b) Resultado obtido para K-means do segundo 111 do terceiro quarto do áudio

Figura 5.7: Resultado obtido no segundo 111 do terceiro quarto do áudio

Tabela 5.1: Tabela de rácio de acerto para GMM num segmento de 20 segundos

| | Total USV | Acertados (GMM) | Rácio (GMM) |
|---|---|---|---|
| Crias | 47 | 42 | 0,86 |
| Progenitora | 4 | 3 | 0,75 |
| Ambas | 9 | 8 | 0,89 |

Tabela 5.2: Tabela de rácio de acerto para K-means num segmento de 20 segundos

| | Total USV | Acertados (K-means) | Rácio (K-means) |
|---|---|---|---|
| Crias | 47 | 33 | 0,70 |
| Progenitora | 4 | 3 | 0,75 |
| Ambas | 9 | 4 | 0,44 |

mais preciso para estas vocalizações. Não se pode esperar sempre este resultado, pois, como visível na Figura 5.6, foi no método de Gaussian Mixture que obteve os resultados mais próximos do pretendido e, na Figura 5.7, o resultado obtido para os dois métodos é semelhante.

Para atingir um resultado mais preciso, foi realizado uma análise de um período de 20 segundos de um sinal. Nas Tabelas 5.1 e 5.2 é possível observar os resultados obtidos. Salienta-se que a leitura foi limitada por ser impraticável fazer a análise completa manualmente. Isto vem do facto de não haver anotações prévias que permitam a sua verificação automática. Desta forma, cada leitura é demorada, não só por ser o utilizador a verificar mas, também, pelo facto que cada visualização necessita que o gráfico atualize; um processo que pode levar alguns minutos ou até mesmo mais que uma hora. As tabelas apresentam o rácio de acerto dos dois métodos, onde se pode verificar qual dos métodos foi mais consistente para cada um dos dados: crias, progenitora ou ambas. Neste caso, salienta-se que o rácio foi, em geral, superior para o método de GMM, sendo superior tanto na cria como para ambas. Além disso, estes dados revelam que existe um maior número de vocalizações realizadas pelas crias.

Desta forma, constata-se que os métodos são capazes de identificar as vocalizações dos murganhos. No entanto, têm dificuldades nos períodos de silêncio. Entre métodos, o método de GMM foi o que teve os resultados mais próximos do pretendido.

# Capítulo 6

# Conclusões

Esta dissertação teve como objetivo o desenvolvimento de um algoritmo que identificasse as vocalizações de murganhos entre uma progenitora e as suas crias. O projeto passou por diversos problemas ao longo da sua execução. Como referido, o *hardware* utilizado mostrou estar aquém do necessário para uma resolução mais eficiente e eficaz. Além disso, a limitação de tempo foi, também, um grande fator para que alguns métodos não fossem tão bem abordados como em outros projetos.

Não obstante, os objetivos foram atinjidos, tendo-se verificado a capacidade de identificar as vocalizações dos murganhos. No entanto, o resultado não foi o ideal, sugerindo que mais pesquisas sejam levadas a cabo. Ao utilizar o programa desenvolvido, será sempre realizado um treino com os dados do ficheiro pretendido, o que leva a que os resultados sejam diferentes para cada áudio testado.

Por outro lado, os outros métodos de *clustering*, presentes na biblioteca e que não puderam ser utilizados, o Spectral Clustering, Agglomerative Clustering e Birtch, podem, também, vir a ser úteis. No entanto, a necessidade de utilizar memória RAM a mais que a possuída pelo computador, pode ser a razão de poucos projetos utilizarem utilizaram-nos.

## 6.1 Trabalho Futuro

Para uma possível continuação desta pesquisa, o próximo passo poderá ser a realização de testes com mais dados. Isto é, testar, para os mesmos métodos, qual o resultado obtido quando se faz *clustering* com um treino com mais ficheiros de

áudio. Na situação de se utilizar um computador com mais memória, poder-se-á testar com os métodos descartados, para se averiguar se têm vantagens em relação aos dois métodos estudados.

# Referências


[1] J. Chabout, A. Sarkar, S. R. Patel, T. Radden, D. B. Dunson, S. E. Fisher, and E. D. Jarvis, "A Foxp2 Mutation Implicated in Human Speech Deficits Alters Sequencing of Ultrasonic Vocalizations in Adult Male Mice," *Frontiers in Behavioral Neuroscience*, vol. 10, 2016. [Citado nas páginas 1, 2, 6, 11, 14 e 15]

[2] J. R. Browning, A. C. Whiteman, L. Y. Leung, X.-C. M. Lu, and D. A. Shear, "Air-puff induced vocalizations: A novel approach to detecting negative affective state following concussion in rats," *Journal of Neuroscience Methods*, vol. 275, pp. 45–49, Jan. 2017. [Citado nas páginas 1, 6, 11, 13, 14 e 16]

[3] D. J. Barker, S. J. Simmons, L. C. Servilio, D. Bercovicz, S. Ma, D. H. Root, A. P. Pawlak, and M. O. West, "Ultrasonic vocalizations: evidence for an affective opponent process during cocaine self-administration," *Psychopharmacology*, vol. 231, pp. 909–918, Mar. 2014. [Citado nas páginas 1, 13 e 14]

[4] K. R. Coffey, R. E. Marx, and J. F. Neumaier, "DeepSqueak: a deep learning-based system for detection and analysis of ultrasonic vocalizations," *Neuropsychopharmacology*, vol. 44, pp. 859–868, Apr. 2019. Number: 5 Publisher: Nature Publishing Group. [Citado nas páginas 1, 2, 6, 11 e 14]

[5] A. M. Ahrens, S. T. Ma, E. Y. Maier, C. L. Duvauchelle, and T. Schallert, "Repeated intravenous amphetamine exposure: Rapid and persistent sensitization of 50-kHz ultrasonic trill calls in rats," *Behavioural Brain Research*, vol. 197, pp. 205–209, Jan. 2009. [Citado nas páginas 1 e 2]

[6] C. Hernandez, M. Sabin, and T. Riede, "Rats concatenate 22 kHz and 50 kHz calls into a single utterance," *Journal of Experimental Biology*, vol. 220, pp. 814–821, Mar. 2017. [Citado nas páginas 2 e 13]

[7] J. M. Wright, J. C. Gourdon, and P. B. S. Clarke, "Identification of multiple call categories within the rich repertoire of adult rat 50-kHz ultrasonic vocalizations: effects of amphetamine and social context," *Psychopharmacology*, vol. 211, pp. 1–13, July 2010. [Citado nas páginas 2, 6, 11, 13 e 14]

[8] M. V. Segbroeck, A. T. Knoll, P. Levitt, and S. Narayanan, "MUPET—Mouse Ultrasonic Profile ExTraction: A Signal Processing Tool for Rapid and Unsupervised Analysis of Ultrasonic Vocalizations," *Neuron*, vol. 94, pp. 465–485.e5, May 2017. Publisher: Elsevier. [Citado nas páginas 2, 7 e 11]

[9] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker Diarization: A Review of Recent Research," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, pp. 356–370, Feb. 2012. Conference Name: IEEE Transactions on Audio, Speech, and Language Processing. [Citado nas páginas 5, 7, 12, 14, 19 e 21]

[10] S. M. Zala, D. Reitschmidt, A. Noll, P. Balazs, and D. J. Penn, "Automatic mouse ultrasound detector (A-MUD): A new tool for processing rodent vocalizations," *PLOS ONE*, vol. 12, p. e0181200, July 2017. Publisher: Public Library of Science. [Citado nas páginas 6, 11, 15 e 16]

[11] Z. D. Burkett, N. F. Day, O. Peñagarikano, D. H. Geschwind, and S. A. White, "VoICE: A semi-automated pipeline for standardizing vocal analysis across models," *Scientific Reports*, vol. 5, p. 10237, May 2015. Publisher: Nature Publishing Group. [Citado nas páginas 7, 11 e 14]

[12] G. Friedland, O. Vinyals, Y. Huang, and C. Muller, "Prosodic and other Long-Term Features for Speaker Diarization," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, pp. 985–993, July 2009. Conference Name: IEEE Transactions on Audio, Speech, and Language Processing. [Citado nas páginas 8, 12, 21 e 22]

[13] E. Shriberg, "Higher-Level Features in Speaker Recognition," in *Speaker Classification I: Fundamentals, Features, and Methods* (C. Müller, ed.), pp. 241–259, Berlin, Heidelberg: Springer, 2007. [Citado nas páginas 8 e 12]

[14] S. Shum, N. Dehak, E. Chuangsuwanich, D. Reynolds, and J. Glass, "Exploiting intra-conversation variability for speaker diarization," in *Interspeech 2011*, pp. 945–948, ISCA, Aug. 2011. [Citado nas páginas 8, 12 e 19]

[15] S. H. Shum, N. Dehak, R. Dehak, and J. R. Glass, "Unsupervised Methods for Speaker Diarization: An Integrated and Iterative Approach," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, pp. 2015–2028, Oct. 2013. Conference Name: IEEE Transactions on Audio, Speech, and Language Processing. [Citado nas páginas 8, 12 e 19]

[16] S. Horiguchi, S. Watanabe, P. Garcia, Y. Xue, Y. Takashima, and Y. Kawaguchi, "Towards Neural Diarization for Unlimited Numbers of Speakers Using

Global and Local Attractors," Sept. 2021. arXiv:2107.01545. [Citado nas páginas 9, 12, 19 e 25]

[17] S. Horiguchi, Y. Fujita, S. Watanabe, Y. Xue, and P. Garcia, "Encoder-Decoder Based Attractors for End-to-End Neural Diarization," Mar. 2022. arXiv:2106.10654. [Citado nas páginas 9, 12, 19 e 25]

[18] K. Yao, M. Bergamasco, M. L. Scattoni, and A. P. Vogel, "A review of ultrasonic vocalizations in mice and how they relate to human speecha)," *The Journal of the Acoustical Society of America*, vol. 154, pp. 650–660, Aug. 2023. [Citado nas páginas 13, 14 e 15]

[19] J. B. Panksepp, K. A. Jochman, J. U. Kim, J. J. Koy, E. D. Wilson, Q. Chen, C. R. Wilson, and G. P. Lahvis, "Affiliative Behavior, Ultrasonic Communication and Social Reward Are Influenced by Genetic Variation in Adolescent Mice," *PLOS ONE*, vol. 2, p. e351, Apr. 2007. Publisher: Public Library of Science. [Citado na página 13]

[20] K. M. Seagraves, B. J. Arthur, and S. E. R. Egnor, "Evidence for an audience effect in mice: male social partners alter the male vocal response to female cues," *Journal of Experimental Biology*, vol. 219, pp. 1437–1448, May 2016. [Citado na página 13]

[21] M. Wöhr and R. K. W. Schwarting, "Affective communication in rodents: ultrasonic vocalizations as a tool for research on emotion and motivation," *Cell and Tissue Research*, vol. 354, pp. 81–97, Oct. 2013. [Citado na página 14]

[22] H. Aleyasin, M. E. Flanigan, S. A. Golden, A. Takahashi, C. Menard, M. L. Pfau, J. Multer, J. Pina, K. A. McCabe, N. Bhatti, G. E. Hodes, M. Heshmati, R. L. Neve, E. J. Nestler, E. A. Heller, and S. J. Russo, "Cell-Type-Specific Role of FosB in Nucleus Accumbens In Modulating Intermale Aggression," *Journal of Neuroscience*, vol. 38, pp. 5913–5924, June 2018. Publisher: Society for Neuroscience Section: Research Articles. [Citado na página 14]

[23] D. Williams, T. Ağabeyoğlu, A. Gafos, and P. Escudero, "Acoustic Similarity Predicts Vowel Phoneme Detection in an Unfamiliar Regional Accent: Evidence from Monolinguals, Bilinguals and Second-Language Learners," *Languages*, vol. 9, p. 62, Feb. 2024. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute. [Citado na página 14]

[24] P. P. Vaidyanathan and B.-J. Yoon, "The role of signal-processing concepts in genomics and proteomics," *Journal of the Franklin Institute*, vol. 341, pp. 111–135, Jan. 2004. [Citado na página 14]

[25] W. J. Tompkins, ed., *BIOMEDICAL DIGITAL SIGNAL PROCESSING C-Language Examples and Laboratory Experiments for the IBM ® PC*. Upper Saddle River, New Jersey: PRENTICE HALL, 2000. [Citado na página 14]

[26] D. Middleton, "Non-Gaussian noise models in signal processing for telecommunications: new methods an results for class A and class B noise models," *IEEE Transactions on Information Theory*, vol. 45, pp. 1129–1149, May 1999. Conference Name: IEEE Transactions on Information Theory. [Citado na página 14]

[27] D. C. Mann, W. T. Fitch, H.-W. Tu, and M. Hoeschele, "Universal principles underlying segmental structures in parrot song and human speech," *Scientific Reports*, vol. 11, p. 776, Jan. 2021. Publisher: Nature Publishing Group. [Citado na página 16]

[28] P. Podder, T. Khan, M. Khan, and M. Rahman, "Comparative performance analysis of hamming, hanning and blackman window," *International Journal of Computer Applications*, vol. 96, pp. 1–7, 06 2014. [Citado na página 16]

[29] W. v. Drongelen, "6 - Continuous, Discrete, and Fast Fourier Transform," in *Signal Processing for Neuroscientists* (W. v. Drongelen, ed.), pp. 91–105, Burlington: Academic Press, 2007. [Citado na página 17]

[30] "Chapter 1 - Multirate Signal Processing," in *Wavelet Analysis and Its Applications* (B. W. Suter, ed.), vol. 8 of *Multirate and Wavelet Signal Processing*, pp. 1–28, Academic Press, Jan. 1998. [Citado na página 17]

[31] Y. Yang, Z. Peng, W. Zhang, and G. Meng, "Parameterised time-frequency analysis methods and their engineering applications: A review of recent advances," *Mechanical Systems and Signal Processing*, vol. 119, pp. 182–221, 2019. [Citado na página 18]

[32] S. Yerabati, "Comprehensive Audio Analysis and Comparison of MFCCs, Spectrograms, CNNs for Speaker Recognition," [Citado na página 19]

[33] T. J. Park, N. Kanda, D. Dimitriadis, K. J. Han, S. Watanabe, and S. Narayanan, "A Review of Speaker Diarization: Recent Advances with Deep Learning," Nov. 2021. arXiv:2101.09624. [Citado nas páginas 19 e 25]

[34] W. Wang, Q. Lin, D. Cai, and M. Li, "Similarity measurement of segment-level speaker embeddings in speaker diarization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 2645–2658, 2022. [Citado na página 19]

[35] H. Fayek, "Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between," Apr. 2016. [Citado nas páginas 19 e 20]

[36] Z. K. Abdul and A. K. Al-Talabani, "Mel Frequency Cepstral Coefficient and its Applications: A Review," *IEEE Access*, vol. 10, pp. 122136–122158, 2022. Conference Name: IEEE Access. [Citado nas páginas 19 e 21]

[37] M. Kotti, V. Moschou, and C. Kotropoulos, "Speaker segmentation and clustering," *Signal Processing*, vol. 88, pp. 1091–1124, May 2008. [Citado nas páginas 19, 20 e 21]

[38] N. S. Ibrahim and D. A. Ramli, "I-vector Extraction for Speaker Recognition Based on Dimensionality Reduction," *Procedia Computer Science*, vol. 126, pp. 1534–1540, Jan. 2018. [Citado na página 25]

[39] W. Wang, W. Song, C. Chen, Z. Zhang, and Y. Xin, "I-vector features and deep neural network modeling for language recognition," *Procedia Computer Science*, vol. 147, pp. 36–43, Jan. 2019. [Citado na página 25]

[40] V. R. Kumar, H. K. Vydana, and A. K. Vuppala, "Significance of GMM-UBM based Modelling for Indian Language Identification," *Procedia Computer Science*, vol. 54, pp. 231–236, Jan. 2015. [Citado na página 25]

[41] M. He, D. Raj, Z. Huang, J. Du, Z. Chen, and S. Watanabe, "Target-speaker Voice Activity Detection with Improved I-Vector Estimation for Unknown Number of Speaker," Aug. 2021. arXiv:2108.03342. [Citado na página 25]

[42] S. Horiguchi, S. Watanabe, P. Garcia, Y. Takashima, and Y. Kawaguchi, "Online Neural Diarization of Unlimited Numbers of Speakers Using Global and Local Attractors," Dec. 2022. arXiv:2206.02432. [Citado na página 25]

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Las Vegas, NV, USA), pp. 770–778, IEEE, June 2016. [Citado na página 26]

[44] F. Landini, J. Profant, M. Diez, and L. Burget, "Bayesian HMM clustering of x-vector sequences (VBx) in speaker diarization: Theory, implementation and analysis on standard tasks," *Computer Speech & Language*, vol. 71, p. 101254, Jan. 2022. [Citado na página 26]

[45] S. Michieletto, F. Stival, and E. Pagello, "Chapter 3 - A probabilistic approach to reconfigurable interactive manufacturing and coil winding for Industry 4.0," in *Advances in Mathematics for Industry 4.0* (M. Ram, ed.), pp. 61–93, Academic Press, Jan. 2021. [Citado na página 26]

[46] P. Wittek, "5 - Unsupervised Learning," in *Quantum Machine Learning* (P. Wittek, ed.), pp. 57–62, Boston: Academic Press, Jan. 2014. [Citado na página 28]

[47] V. Kotu and B. Deshpande, "Chapter 4 - Classification," in *Data Science (Second Edition)* (V. Kotu and B. Deshpande, eds.), pp. 65–163, Morgan Kaufmann, Jan. 2019. [Citado na página 29]

[48] M. Khosla, K. Jamison, G. H. Ngo, A. Kuceyeski, and M. R. Sabuncu, "Machine learning in resting-state fMRI analysis," *Magnetic Resonance Imaging*, vol. 64, pp. 101–121, Dec. 2019. [Citado na página 29]

[49] R. Ranjbarzadeh, S. Dorosti, S. Jafarzadeh Ghoushchi, A. Caputo, E. B. Tirkolaee, S. S. Ali, Z. Arshadi, and M. Bendechache, "Breast tumor localization and segmentation using machine learning techniques: Overview of datasets, findings, and methods," *Computers in Biology and Medicine*, vol. 152, p. 106443, Jan. 2023. [Citado na página 29]

[50] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. [Citado na página 32]

[51] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. [Citado na página 32]

[52] J. Fischer and K. Hammerschmidt, "Ultrasonic vocalizations in mouse models for speech and socio-cognitive disorders: insights into the evolution of vocal communication," *Genes, Brain, and Behavior*, vol. 10, pp. 17–27, Feb. 2011. [Citado na página 33]

[53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. [Citado na página 33]

[54] B. McFee, M. McVicar, D. Faronbi, I. Roman, M. Gover, S. Balke, S. Seyfarth, A. Malek, C. Raffel, V. Lostanlen, B. v. Niekirk, D. Lee, F. Cwitkowitz, F. Zalkow, O. Nieto, D. Ellis, J. Mason, K. Lee, B. Steers, E. Halvachs, C. Thomé,

F. Robert-Stöter, R. Bittner, Z. Wei, A. Weiss, E. Battenberg, K. Choi, R. Yamamoto, C. J. Carr, A. Metsai, S. Sullivan, P. Friesch, A. Krishnakumar, S. Hidaka, S. Kowalik, F. Keller, D. Mazur, A. Chabot-Leclerc, C. Hawthorne, C. Ramaprasad, M. Keum, J. Gomez, W. Monroe, V. A. Morozov, K. Eliasi, nullmightybofo, P. Biberstein, N. D. Sergin, R. Hennequin, R. Naktinis, beantowel, T. Kim, J. P. Åsen, J. Lim, A. Malins, D. Hereñú, S. v. d. Struijk, L. Nickel, J. Wu, Z. Wang, T. Gates, M. Vollrath, A. Sarroff, Xiao-Ming, A. Porter, S. Kranzler, Voodoohop, M. D. Gangi, H. Jinoz, C. Guerrero, A. Mazhar, toddrme2178, Z. Baratz, A. Kostin, X. Zhuang, C. T. Lo, P. Campr, E. Semeniuc, M. Biswal, S. Moura, P. Brossier, H. Lee, and W. Pimenta, "librosa/librosa: 0.10.2.post1," May 2024. [Citado na página 33]

[55] F. Pérez and B. E. Granger, "IPython: a system for interactive scientific computing," *Computing in Science and Engineering*, vol. 9, pp. 21–29, May 2007. [Citado na página 34]