



UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA E ELETRÔNICA

Speaker Diarization Utilizando LSTM e Spectral Clustering

Trabalho de Conclusão de Curso apresentado ao curso de Engenharia Eletrônica, Departamento de Engenharia Elétrica e Eletrônica da Universidade Federal de Santa Catarina como requisito parcial para obtenção do grau de bacharel no Curso de Engenharia Eletrônica.

Bruno Griep Fernandes
Orientador: Richard Demo Souza

Florianópolis, 12 de julho de 2019.

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Fernandes, Bruno Griep
Speaker Diarization Utilizando LSTM e Spectral
Clustering / Bruno Griep Fernandes ; orientador,
Richard Demo Souza, 2019.
90 p.

Trabalho de Conclusão de Curso (graduação) -
Universidade Federal de Santa Catarina, Centro
Tecnológico, Graduação em Engenharia Eletrônica,
Florianópolis, 2019.

Inclui referências.

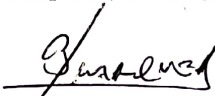
1. Engenharia Eletrônica. 2. Processamento de
Fala. 3. Machine Learning. 4. Speaker Diarization.
5. Deep Learning. I. Souza, Richard Demo. II.
Universidade Federal de Santa Catarina. Graduação em
Engenharia Eletrônica. III. Título.

Bruno Griep Fernandes

**SPEAKER DIARIZATION UTILIZANDO LSTM E
SPECTRAL CLUSTERING**

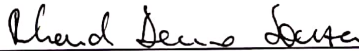
Este Trabalho de Conclusão de Curso foi julgado adequado no contexto da disciplina EEL7806 Projeto Final TCC, e aprovado em sua forma final pelo Departamento de Engenharia Elétrica e Eletrônica da Universidade Federal de Santa Catarina.

Florianópolis, 12 de julho de 2019.



Prof. Jefferson Luiz Brum Marques, PhD.

Coordenador do Curso

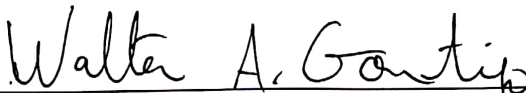


Prof. Richard Demo Souza, D.Sc.

Orientador

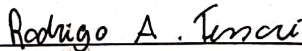
Universidade Federal de Santa Catarina

Banca examinadora:



Walter Antônio Gontijo, M.Sc.

Universidade Federal de Santa Catarina



Eng. Rodrigo Alexandre Salerno del Menezzi Tessari

Universidade Federal de Santa Catarina

*Dedico este trabalho à minha mãe e ao meu pai, pelo apoio durante
toda minha graduação.*

Agradecimentos

Desejo expressar meu reconhecimento a todos que, de uma maneira ou outra, colaboraram na realização deste trabalho.

Agradeço à minha família por todo o apoio, estabilidade e motivação durante minha graduação.

Ao Laboratório de Circuitos e Processamento de Sinais (LINSE) pelas oportunidades oferecidas.

Ao Walter Antônio Gontijo, pelos anos de supervisão e aprendizado proporcionados. Aos colegas do LINSE, em especial ao André, Daniel, Hermeson e Rodrigo pelas discussões e sugestões que possibilitaram o desenvolvimento deste trabalho.

I like things to happen. And if they don't happen, I like to make them happen. (Winston Churchill)

RESUMO

Este trabalho apresenta um sistema de *Speaker Diarization*. Em aplicações envolvendo este sistema, deve-se marcar em um canal de áudio os trechos de fala correspondentes a determinados locutores. Neste trabalho são apresentadas uma breve revisão do estado da arte, uma implementação do método considerado o estado da arte em *Speaker Diarization*, como também uma avaliação de desempenho do sistema implementado.

Palavras-chave: *Speaker Diarization*. Aprendizagem de Máquina. Processamento de Fala.

ABSTRACT

This work presents a Speaker Diarization system. In applications involving this system, the speakers in an audio channel should be annotated in every speech segment corresponding to a specific speaker. In this work are presented a brief state of the art review, an implementation using the state of the art method for Speaker Diarization, as well as a performance evaluation for the implemented system.

Keywords: Speaker Diarization. Machine Learning. Speech Processing.

Lista de Figuras

1.1	Exemplo de Aplicação: Aumento da fidelidade de um ASR.	2
1.2	Marcação de dois Locutores via <i>Speaker Diarization</i> .	2
1.3	Diagrama de Blocos de um Sistema de <i>Speaker Diarization</i>	3
1.4	Marcação dos instantes de tempo de mudança de locutor em contraste às marcações dos locutores.	4
2.1	Diagrama de Blocos Genérico de um Sistema de <i>Speaker Diarization</i>	10
3.1	Diagrama de Blocos do Sistema de <i>Speaker Diarization</i> Implementado	19
3.2	Representação gráfica dos passos de refinamento na matriz de similaridades.	22
3.3	Marcações de Referência e Hipótese gerada pelo sistema <i>Speaker Diarization</i> implementado.	24
3.4	Histogramas referentes às métricas do sistema no Cenário 1.	27
3.5	Histogramas referentes às métricas do sistema no Cenário 2.	28
A.1	Janela de Hanning com tamanho 51 amostras.	44
A.2	Efeitos do <i>Overlap</i> sobre sinais janelados.	45
A.3	Sinal de fala e sua representação em espectrograma.	46
A.4	Representação gráfica do método <i>overlap-add</i>	48

A.5	Relação entre escala linear e escala <i>mel</i> de frequência. . . .	49
A.6	Diagrama de Blocos do cálculo do <i>Mel-frequency Cepstrum Coefficients</i>	50
A.7	Resposta em frequências do banco de filtros de 13 bandas <i>mel</i>	51
A.8	Parâmetros MFCC de um sinal de fala.	51
B.1	Classificação Binária.	55
B.2	<i>K-means Clustering</i> com $K = 2$	56
B.3	Representação gráfica da convergência da função custo J	57
B.4	Comparação entre <i>K-means</i> e <i>Spectral Clustering</i>	58
B.5	Exemplo de uma rede neural MLP com duas camadas ocultas.	61
B.6	Rede Neural Recorrente com laço desenrolado.	62
B.7	Tipos de Arquiteturas de Redes Neurais Recorrentes.	64
B.8	Topologia da rede LSTM.	65

Lista de Tabelas

2.1	Diferenças entre os tipos de gravações.	9
2.2	Comparativo entre métodos de <i>Speaker Diarization</i>	16
3.1	Estrutura da Rede LSTM.	20
3.2	Lista com os 17 idiomas presentes na base MUSAN.	25
3.3	Cenário com locutores de países distintos	25
3.4	Cenário com dois locutores Americanos.	26
C.1	Divisão em <i>subsets</i> da base LibriSpeech	68
C.2	Estatísticas das bases VoxCeleb1 e VoxCeleb2.	69

Sumário

1	Introdução	1
1.1	O que não é <i>Speaker Diarization</i> ?	3
1.1.1	Reconhecimento de Locutor (<i>Speaker Recognition</i>)	3
1.1.2	Mudança de Locutor (<i>Speaker Change Detection</i>)	3
1.1.3	Separação de Fontes (<i>Source Separation</i>)	4
1.2	Objetivos	4
1.3	Estrutura do Trabalho	5
2	Revisão do Estado da Arte	7
2.1	Aplicações e Desafios	8
2.2	Arquitetura Básica e Metodologia	9
2.3	Extração de parâmetros	10
2.4	Detecção Ativa de Fala e Segmentação	12
2.5	<i>Clustering</i>	12
2.5.1	<i>Bottom-Up</i>	13
2.5.2	<i>Top-Down</i>	13
2.6	Novas Abordagens e Estado da Arte	14
2.6.1	Modelo de Mistura de Gaussianas	14
2.6.2	A Representação i-vector	14
2.7	Novas Abordagens e Comparações	15

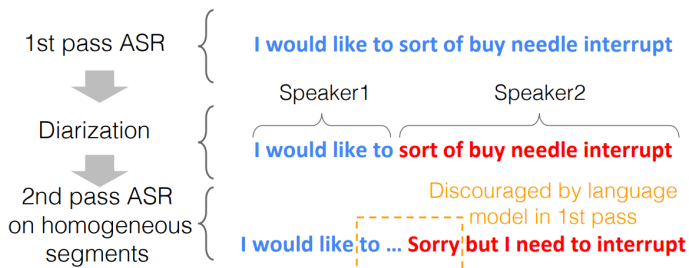
3	Implementação e Avaliação do Sistema	17
3.1	Sistema de <i>Speaker Diarization</i>	18
3.2	Segmentação ou Pré-processamento	18
3.3	Rede Pré-treinada	18
3.4	Etapa de <i>Clustering</i>	21
3.5	Avaliação do Sistema	23
3.6	Cenários de Avaliação	24
3.7	Resultados e Discussões	25
3.7.1	Comparativo entre Sistema Base e Implementado	26
3.8	Tendências Futuras	26
4	Conclusão	29
	Referências Bibliográficas	31
A	Conceitos Básicos de Processamento Digital de Sinais	41
A.1	Transformada de Fourier e Espectrograma	42
A.2	Método <i>Overlap-Add</i>	46
A.3	Escala mel	48
B	Conceitos Básicos de Aprendizagem de Máquina	53
B.1	Categorias de Aprendizagem de Máquina	53
B.2	Sub-Categorias de Aprendizagem Supervisionada	54
B.2.1	Classificação	54
B.3	Sub-Categorias de Aprendizagem Não Supervisionada	54
B.3.1	<i>K-means</i>	54
B.3.2	<i>Spectral Clustering</i>	57
B.4	<i>Deep Learning</i>	59
B.4.1	<i>Multilayer Perceptrons</i> (MLPs)	59
B.4.2	Funções de Ativação	60
B.4.3	Redes Neurais Recorrentes (RNNs)	61
B.4.4	Extração de Parâmetros	65
C	Bases de Dados Utilizadas	67
C.1	VCTK Corpus	67
C.2	LibriSpeech ASR Corpus	68
C.3	VoxCeleb	68
C.4	MUSAN: <i>A Music, Speech, and Noise Corpus</i>	69

CAPÍTULO 1

Introdução

Speaker Diarization é o sistema utilizado para responder “quem fala quando” [3]. Em um canal de áudio com conteúdo de mais de um locutor, como por exemplo gravações de conversas em reuniões. O conhecimento de “quem fala quando” possibilita aplicações como a transcrição automática de fala para texto em consultórios, visando melhorar a comunicação entre médicos e pacientes [71]; separação de locutores via parâmetros áudio visuais [70]; análise de conversas entre cliente e atendente em *call centers*. A Figura 1.1 ilustra a aplicação de *Speaker Diarization* para aumentar a fidelidade em reconhecedores automáticos de locutores (ASR), minimizando erros provenientes de modelos linguísticos.

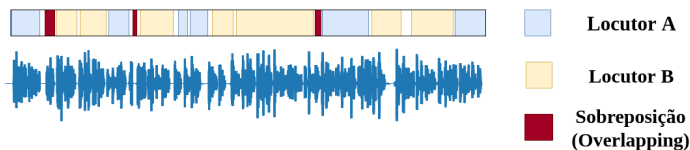
Figura 1.1: Exemplo de Aplicação: Aumento da fidelidade de um ASR.



Fonte: Quang Wang em [66].

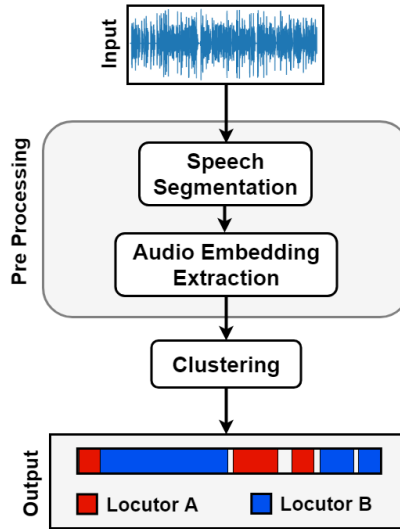
Para obter-se a resposta dada pelo *Speaker Diarization* é necessário demarcar neste canal de áudio as ocorrências de fala de cada locutor. A Figura 1.2 apresenta um caso de marcação para um canal de áudio com apenas dois locutores. Note que existem trechos com possível sobreposição entre dois locutores.

Figura 1.2: Marcação de dois Locutores via *Speaker Diarization*.



Fonte: do Autor.

O processo de *Speaker Diarization* usualmente consiste em quatro etapas [3, 27, 61, 67], representadas pelo diagrama de blocos da Figura 1.3, sendo elas: *i*) Segmentação da fala, em que o sinal é segmentado em pequenas janelas de tempo, e as partes sem fala são filtradas; *ii*) *Audio embedding extration*, processo em que parâmetros específicos são extraídos, tais como MFCCs, FFT e espectrograma; *iii*) *Clustering*: o número de locutores é determinado assim como a qual locutor pertence cada *feature* extraída; *iv*) Resegmentação, que refina resultados da etapa de *clustering* de forma a agrupar as marcações de cada locutor.

Figura 1.3: Diagrama de Blocos de um Sistema de *Speaker Diarization*

Fonte: do Autor.

1.1 O que não é *Speaker Diarization*?

Speaker Diarization é facilmente confundido com outras técnicas, e esta seção procura deixar claro o que *Speaker Diarization* não é.

1.1.1 Reconhecimento de Locutor (*Speaker Recognition*)

É o sistema usado para responder “quem está falando”. Retorna a identificação de certo indivíduo apenas utilizando suas características vocais [14].

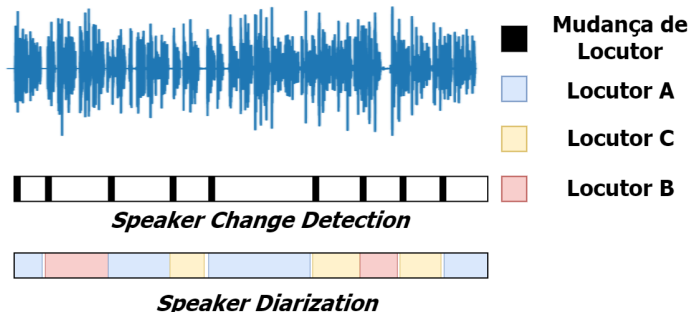
Portanto, diferentemente do *Speaker Diarization*, que retorna as marcações de mesmo locutor em um canal de áudio, *Speaker Recognition* determina que locutor está presente no áudio.

1.1.2 Mudança de Locutor (*Speaker Change Detection*)

É o processo que marca em qual instante de tempo ocorre a mudança de locutor [8, 18]. *Speaker Diarization* por outro lado, determina um

identificador genérico para cada locutor que surge nos pontos de mudança. A Figura 1.4 apresenta a representação gráfica dos dois sistemas para um mesmo sinal de áudio.

Figura 1.4: Marcação dos instantes de tempo de mudança de locutor em contraste às marcações dos locutores.



Fonte: do Autor.

1.1.3 Separação de Fontes (*Source Separation*)

Tal processo obtém ou recupera as componentes originais de cada fonte em um canal de áudio contendo múltiplas fontes combinadas ou sobrepostas (*overlapping*) [43]. A saída de um sistema de separação de fontes contém um canal de áudio exclusivo para cada fonte recuperada.

Diferentemente, no *Speaker Diarization* assume-se que os áudios não estão sobrepostos, e preocupa-se apenas com a marcação das fontes, dessa forma a saída continua sendo apenas um canal de áudio, devidamente marcado.

1.2 Objetivos

O objetivo principal deste trabalho é dominar a técnica de *Speaker Diarization*. Como objetivos específicos temos:

- Apresentar uma breve revisão do estado da arte;
- Implementar um método de *Speaker Diarization*;
- Apresentar possíveis tendências futuras na área.

1.3 Estrutura do Trabalho

O restante deste trabalho está organizado da seguinte forma. O Capítulo 2 apresenta uma breve revisão do estado da arte sobre *Speaker Diarization*. No Capítulo 3 é apresentada uma reprodução de um sistema de *Speaker Diarization* recentemente proposto, além de comparações com outras implementações. Por fim, o Capítulo 4 contém as considerações finais.

CAPÍTULO 2

Revisão do Estado da Arte

Este capítulo contempla uma breve revisão do estado da arte de *Speaker Diarization*, apresentando uma breve introdução histórica além de alguns dos métodos mais utilizados e atuais.¹

O avanço científico em *Speaker Diarization* abrange diferentes domínios, dependendo do foco da pesquisa ou projeto. Uma das primeiras e principais linhas de pesquisas envolviam dados de telefonia e noticiários. No final dos anos 90 e início dos anos 2000 os sistemas eram utilizados para fazer anotações automáticas de transmissões de TV e rádio [3,61]. Essas anotações incluíam a transcrição automática de fala, além da geração de meta dados com marcação dos locutores correspondentes.

A partir de 2002 houve um grande aumento no interesse em aplicações com dados de reuniões, palestras, congressos e similares. Devido a este crescimento, o *US National Institute for Standards and Technology* (NIST) começou a organizar conferências para pesquisas nesta área [3,26], fomentando novas ideias que contribuíssem no avanço do desenvolvimento do estado da arte do tema. O termo *Speaker Diarization*, até então, não era designado para a tarefa de “quem fala quando”, – os

¹Alguns conceitos básicos de processamento digital de sinais e aprendizado de máquina são utilizados durante o capítulo, o leitor que não estiver completamente familiarizado com os termos pode fazer uso dos Apêndices A e B.

sistemas até então eram conhecidos como *Speaker Segmentation* [56] e *Speaker Clustering* [2]. O termo *Speaker Diarization* foi definido apenas em 2004 [35], sendo proposto como uma nova tarefa de processamento de fala, sendo incluído como um tópico de pesquisa nos congressos organizados pelo NIST.

2.1 Aplicações e Desafios

Para dados de noticiários, debates, conferências, filmes, conversas de telefone, consultas médicas, ligações de *call centers*, entre outros que incluam mais de um único locutor, um sistema de *Speaker Diarization* é aconselhável para melhoria de aplicações como *Speaker Verification* e *Speech* ou *Speaker Recognition* [4]. Algumas aplicações mais específicas incluem:

- Marcação de fala e locutor [10];
- Reconhecimento de locutores para casos com mais de um locutor no canal de áudio [65];
- Transcrição automática de noticiários [49];
- Transcrição de fala para texto [54];
- Marcação de locutores em noticiários com informações Audio-Visuais [70];
- Multimodal (mais de um canal de áudio) *Speaker Diarization* [38, 42].

Segundo a literatura, os problemas envolvendo *Speaker Diarization* estão relacionados a três tipos de áudios: *i)* Conversas de telefone, com um único canal de áudio contendo entre dois ou mais locutores com baixa qualidade de gravação; *ii)* Noticiários, programas de rádio ou jornais de TV com conteúdos de debates, conversas, entrevistas e até mesmo comerciais, todos gravados com qualidade superior também em um único canal de áudio. *iii)* Conferências ou locais onde um número de pessoas interagem entre si. As gravações desses dados são obtidas de várias maneiras, por vezes são utilizados arranjos de microfones espalhados no local, outras vezes um único microfone é utilizado. Também utiliza-se gravações com microfones de lapela, que geram um canal de

Tabela 2.1: Diferenças entre os tipos de gravações.

	Conversas de Telefone	Noticiários	Conferências
Número de Locutores	Conhecido (dois ou três)	Desconhecido, porém alto	Desconhecido, limitado pelo tamanho do local
Comprimento dos Segmentos	Usualmente curto	Usualmente longo	Geralmente muito curtos
Mudança de Locutor	Média	Baixa	Alta
Tipo de Trechos sem Fala	Ruído	Ruído, música, comerciais	Vários impulsos ruidosos
Sobreposição de Fala	Média	Baixa	Alta
Mídia de Gravação	Telefone fixo ou celular (banda estreita)	Microfone de Headset/Lapela (banda larga)	Microfones de mesa (banda larga)
Falta na Fluência da Fala	Maior parte do tempo	Raramente	Ocasionalmente

Fonte: *An Overview of Speaker Diarization: Approaches, Resources and Challenges* [4].

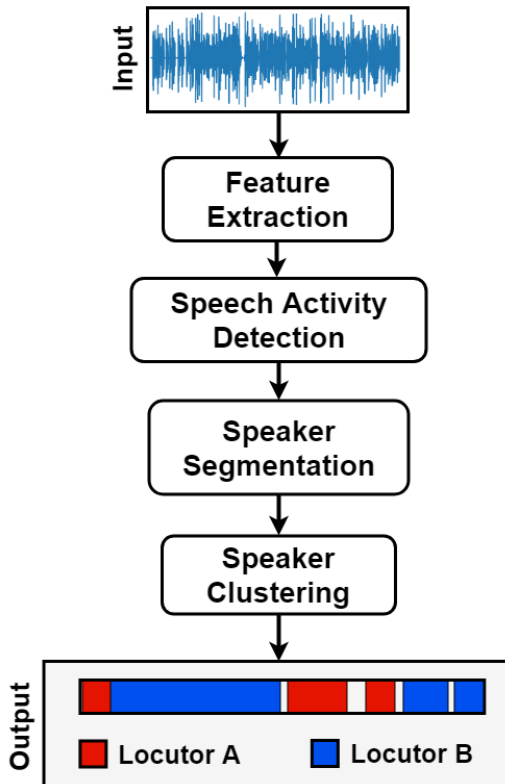
áudio distinto para cada locutor no local, ou seja, cada microfone corresponde a um locutor.

Os desafios são distintos dependendo do tipo de dados utilizados para o desenvolvimento, devido o tipo de gravação ter influência direta na complexidade do problema. A Tabela 2.1 apresenta as diferenças e características dos tipos de dados utilizados em pesquisas na área de *Speaker Diarization*, apresentando suas características. A implementação com dados de conferências se demonstra mais desafiadora que os demais, por possuir variações de ambiente, tipo de gravação, além de trechos de fala espontâneos.

2.2 Arquitetura Básica e Metodologia

A grande maioria dos sistemas são divididos em sub tarefas. Nesta seção serão apresentadas as tarefas utilizadas para a construção do sistema de *Speaker Diarization*, juntamente com a evolução de cada uma das etapas ao longo dos anos. A Figura 2.1 apresenta o diagrama de blocos de um protótipo utilizado em grande parte das implementações, cada bloco corresponde a uma tarefa chave do sistema [4].

Figura 2.1: Diagrama de Blocos Genérico de um Sistema de *Speaker Diarization*



Fonte: do Autor.

2.3 Extração de parâmetros

A extração de características acústicas de um sinal de fala é um dos passos fundamentais para a identificação e caracterização de um locutor. Uma abordagem intuitiva é a análise do espectrograma desses sinais, porém esta análise apresenta inconsistências em extrair informações da fala como a entonação, o ritmo, o acento (intensidade, altura, duração) e demais atributos correlatos na fala.

Um estudo feito em [16] apresentava uma investigação sistemática

de parâmetros para diferenciar diferentes locutores, o estudo fez uso de 70 diferentes parâmetros de longa duração, diferente do espectrograma que apresenta parâmetros de curta duração. Os autores demonstram que ao utilizar parâmetros de longa duração, se tem grande ganho em diminuir as taxas de erros, tendo o MFCC (*Mel-frequency Cepstrum Coefficients*) apresentando o melhor desempenho. A motivação do estudo deve-se ao fato de que na pesquisa em *Speaker Recognition* havia grande dominância do uso de parâmetros de curta duração até o ano de 2007, sendo assim os autores decidiram investigar diferentes abordagens e comparar seus resultados.

Foi sugerido em [55], a consideração de padrões de longa duração, como apresentado em [16], pois estes padrões revelavam características individuais de cada segmento de fala de um locutor, como seu comportamento, emoções entre outras características que não eram possíveis extrair com parâmetros de curta duração. Ainda em [55] efetuou-se uma análise de desempenho dos sistemas de *Speaker Recognition*, apresentados no congresso NIST RT 2007 que utilizavam MFCCs como parâmetro de análise. Estes sistemas lideravam com uma taxa relativa de erro de diarização (DER) de 30%, que para sistemas que não utilizavam MFCCs a taxa era em torno de 67%.

O uso de outros parâmetros cepstrais, como também MFCCs de tamanhos diferentes, são recorrentes na extração de informações de locutores em tarefas de *Speaker Diarization*. No sistema LIUM [33] durante a etapa de segmentação são utilizados MFCCs com número de coeficientes entre 12 a 19, e para a etapa de clusterização são utilizadas as derivadas de primeira e segunda ordem dos coeficientes. Algumas implementações apresentam o uso de LFCCs [63] (*Linear Prediction Cepstral Coefficients*), extraídos de forma similar aos MFCCs porém o espaçamento entre os filtros é linear, contrastando com o banco de filtros usados nos MFCCs, que estão espaçados de acordo com a escala *mel*. Porém a implementação utilizando LFCCs não demonstrou de forma conclusiva ser superior aos MFCCs, não sendo tão utilizada. Ambos trabalhos citados utilizavam análises com janelas de 25 a 30 ms de duração, com salto entre janelas de 10 ms.

Recentemente, com o uso de redes neurais houve um grande aumento de performance dos sistemas de *Speaker Diarization*. Ao utilizar parâmetros retirados da última camada de ativação de uma rede neu-

ral é possível obter-se uma nova representação chamada de *d-vector*. Em [51] é demonstrada uma melhoria de 50% para a detecção de fala na base de dados do YouTube, ao usar uma nova representação de parâmetros gerados por uma rede neural profunda com 13 MFCCs como entrada da rede. Neste trabalho são utilizados parâmetros extraídos de uma rede neural LSTM com 40 MFCCs como entrada da rede, metodologia esta baseada em [66].

2.4 Detecção Ativa de Fala e Segmentação

A etapa de detecção ativa de fala (*Speech Activity Detection*) marca regiões onde há fala em um canal de áudio, ignorando trechos com silêncio ou ruído. Os métodos mais utilizados e fundamentados na literatura usam classificação por máxima verossimilhança (*maximum-likelihood*) com GMMs (*Gaussian Mixture Models*) de forma supervisionada [4], para conseguir diferenciar um trecho de fala de um trecho ruidoso por exemplo.

A etapa de segmentação tem o objetivo de dividir um sinal em pequenos segmentos de mesmo tamanho, possibilitando detectar a mudança de locutor ao usar técnicas de comparação de semelhanças entre os segmentos. Essa metodologia fundamenta duas linhas de pesquisas: *Speaker Change Detection* [18] e *Speaker Diarization* [65].

2.5 Clustering

Clustering é um problema comum em análise de dados estatísticos, sendo comumente utilizado em diferentes áreas de conhecimento. A tarefa de *clustering* pode ser definida como o processo de agrupamento de objetos que possuem características similares, cada grupo criado para caracterizar esses objetos é chamado de *cluster*. Os objetos podem ser pontos em um espaço vetorial como também modelos estatísticos.

Para o problema de *Speaker Diarization* o objetivo é aplicar algum algoritmo de *clustering* nos segmentos de um áudio com mais de um locutor, e determinar um *cluster* específico para cada segmento. Cada *cluster* determinado pelo algoritmo idealmente deve representar um único locutor, dessa forma o número total de *clusters* de um sistema é determinado pelo número de locutores em um canal de áudio.

As aplicações de *clustering* são divididas em duas abordagens sendo elas denominadas como *bottom-up* e *top-down*. Ambas abordagens são usualmente baseadas em Modelos Ocultos de Markov (*Hidden Markov Models* ou HMMs) onde cada estado é um GMM que corresponde a um locutor [3].

2.5.1 *Bottom-Up*

Também conhecido como o algoritmo AHC (*Agglomerative Hierarchical Clustering*) é a abordagem de *clustering* mais utilizadas na literatura para a tarefa de *diarization* [3, 61]. A abordagem *bottom-up* treina um número grande de *clusters* ou modelos, o número de *clusters* reduz a cada iteração do algoritmo, de tal forma que resta apenas um único *cluster* para cada locutor.

O algoritmo AHC ou *bottom-up* consiste nos seguintes passos:

- (i) Inicialização dos *clusters* para cada segmento de fala;
- (ii) Computa distância entre pares de *clusters*;
- (iii) Unifica *clusters* próximos;
- (iv) Atualiza as distâncias calculadas para os *clusters* restantes;
- (v) Repete os passos *ii* a *iv* até o número de *clusters* ser o mesmo que o de locutores.

2.5.2 *Top-Down*

A abordagem *top-down* em contraste com a *bottom-up* primeiramente modela o canal de áudio como um único locutor e sucessivamente adiciona novos modelos até atingir o número de locutores desejável. Nessa abordagem os *clusters* usualmente são modelados com único GMM para todos segmentos de fala, gradativamente são adicionados novos modelos GMM no treinamento, até obter-se um número suficiente para representar o total de locutores nos segmentos de fala [3].

Diversos critérios de parada do treinamento dos GMMs foram estudados para melhoria de desempenhos dos modelos. Um dos possíveis critérios utilizavam abordagens com limiar de decisão tais como BIC

(*Bayesian Inference Criterion*) [47], Kullback-Leibler (KL) [56] e *Generalized Likelihood Ratio* (GLR) [50]. Um estudo comparativo entre as abordagens *bottom-up* e *top-down* é realizado em [15].

2.6 Novas Abordagens e Estado da Arte

Alguns modelos comumente utilizados para tarefas de *Speaker Verification* e *Speaker Recognition* são adotados para capturar informações de locutores em segmentos de áudio, sendo esses modelos reaproveitados na *Speaker Diarization*. Os dois principais modelos de locutor utilizados são GMMs e *i-vectors*, sendo essa combinação considerada o estado da arte em *Speaker Diarization* até recentemente.

2.6.1 Modelo de Mistura de Gaussianas

O modelo de mistura de gaussianas (GMM) de parâmetros cepstrais é dado por pela PDF (*Probability Density Function*) $p(x)$ definida como

$$p(x) = \sum_{i=0}^N w_i \mathcal{N}(\mu_i, \Sigma_i, x) \quad \text{tal que} \quad \sum_{i=0}^N w_i = 1, \quad (2.1)$$

onde x corresponde a um ponto de um vetor de parâmetros cepstrais, correspondendo a um segmento de fala de um locutor; w_i são os pesos, μ_i as médias estimadas e Σ_i a matriz de covariância do modelo.

Se a duração dos segmentos for pequena, o número disponível de vetores de parâmetros para um segmento é insuficiente para estimar um modelo completo. Para contornar este problema utiliza-se um modelo UBM (*Universal Background Model*) pré-treinado para obter-se um modelo de locutor para cada segmento de fala [25].

Em alguns experimentos envolvendo *Speaker Verification* e *Speaker Recognition* notou-se que apenas a média das Gaussianas de um GMM continha a informação de maior relevância de um locutor [48]. Ao concatenar essas médias cria-se um único vetor (chamado de GMM *supervector*) em um espaço vetorial de grande dimensionalidade.

2.6.2 A Representação *i-vector*

O conceito de *i-vector* foi introduzido em sistemas de *Speaker Verification* como um parâmetro extraído de GMMs, com o intuito de reduzir

a dimensionalidade dos hiperparâmetros de um GMM. Devido o tamanho dos modelos UBM serem da ordem de 512, 1024 ou até mesmo 2048 Gaussianas, o tamanho dos *supervectors* tornaram-se muito grandes para serem computados. Portanto uma nova representação que reduzisse a dimensionalidade desses *supervectors* se tornou necessária.

O *i-vector* portanto foi proposto como uma subespaço pertencente ao espaço vetorial desses *supervectors*, reduzindo a dimensionalidade para mais de centenas de vezes. A representação formal deste novo vetor é dada por

$$m = M + Tx, \quad (2.2)$$

onde m é a média do *supervisor* adaptado para cada *i-vector* x buscado, M é a média do *supervisor* e T é uma matriz que representa a variabilidade total do novo subespaço criado [14], sendo responsável pela redução de dimensionalidade de x ao contrair este *supervisor* em um novo subespaço de vetores.

2.7 Novas Abordagens e Comparações

Durante muitos anos a técnica mais dominante em *Speaker Verification* e *Speaker Diarization* foi a parametrização utilizando *i-vectors*. Entretanto, com o crescimento de *deep learning* em diversas áreas de pesquisa possibilitou uma nova parametrização de características de locutores, conhecida como *d-vectors*. Em diversas áreas que envolvam processamento de fala esse novo método vem superando os sistemas até então considerados como estado da arte.

Em 2018 o Google desenvolveu uma nova abordagem de *Speaker Diarization* baseada em *d-vectors* (*deep vectors*), fazendo uso de uma rede neural LSTM para a retirada desses vetores. Esse sistema apresentou resultados significativos quando comparados com os melhores sistemas baseados em *i-vector* até então [66]. A Tabela 2.2 apresenta a comparação do novo sistema implementado pelo Google, em contraste com dois dos mais famosos sistemas, o LIUM [33] feito em língua inglesa, e o REPERE [17] em língua francesa.

Por ser o novo estado da arte na área de *Speaker Diarization* o sistema do Google foi escolhido como base para reprodução deste trabalho, utilizando as etapas descritas em [66] como base de reprodução, apresentada no Capítulo 3.

Tabela 2.2: Comparativo entre métodos de *Speaker Diarization*

Corporação	Técnica Utilizada	DER (%)
LIUM	GMM com <i>clustering</i> CLR	17.19
REPERE	<i>i-vector</i> com ILP <i>clustering</i>	15.46
Google	<i>d-vector</i> com <i>Spectral Clustering</i>	12.30

Fonte: do Autor.

Implementação e Avaliação do Sistema

Neste capítulo é apresentada uma implementação de *Speaker Diarization*, baseada no artigo “*Speaker Diarization with LSTM*” considerado como o estado da arte em 2018. Também são apresentadas avaliações e comparações com o sistema base. A implementação foi desenvolvida em linguagem de programação PYTHON; a extração de atributos do sinal de áudio foi realizada com o auxílio da biblioteca LibROSA [32], que possui funções de análise de sinais, tais como espectrogramas e MFCCs. Para a definição da arquitetura do modelo da rede neural, assim como seu treinamento, utilizou-se o *framework* TensorFlow [1], que permite a definição e execução de algoritmos de aprendizagem profunda, além de ser compatível com GPUs, diminuindo o tempo computacional total.

Estrutura da Implementação

O sistema é composto por 3 etapas, sendo elas: *i*) Segmentação, onde o sinal de fala é particionado em janelas de mesmo tamanho; *ii*) Extração de parâmetros, onde são calculados os MFCCs dos trechos janelados, usados como entradas de uma LSTM, com o propósito de extrair os parâmetros *d-vectors* a partir dos coeficientes MFCCs; *iii*) *Clustering*,

passo onde infere-se um locutor para cada trecho de fala janelado, a partir dos *d-vectors* retirados da LSTM.

Para a extração de parâmetros, utilizou-se um modelo pré-treinado disponível em [30]. Este modelo pré-treinado corresponde a uma rede para sistemas de *Speaker Verification* e *Recognition*. O processo de treinamento do modelo é apresentado neste trabalho, seguindo os passos do desenvolvedor da rede (Dalon Lobo [30]).

3.1 Sistema de *Speaker Diarization*

Um sistema de *Speaker Diarization* apresenta três etapas: Segmentação de fala; extração de parâmetros e *clustering*. A Figura 3.1 mostra graficamente, em detalhes, as etapas necessárias para a implementação do *Speaker Diarization*. A segmentação de fala é responsável pelo janelamento do sinal de fala, como também o cálculo dos respectivos MFCC; a extração de parâmetros usa os MFCCs como entrada da rede neural LSTM, que gera como saída um vetor de parâmetros. Por fim estes vetores são aglomerados em novos segmentos que passam pela etapa de *clustering*, retornando a qual locutor cada segmento pertence.

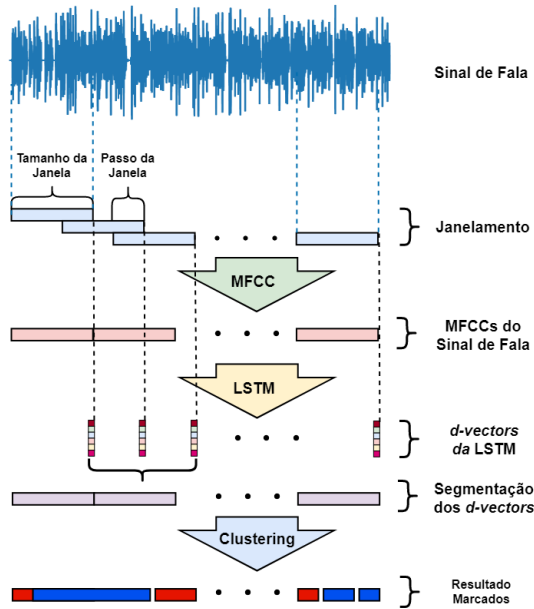
3.2 Segmentação ou Pré-processamento

De forma similar ao sistema base [66], a etapa de segmentação pode ser considerada como um pré-processamento do sinal de fala. Em tal etapa, primeiramente aplica-se um VAD no sinal de fala, sendo escolhido o **webrtcvad** [52] para a tarefa. Logo após, divide-se um dado sinal de áudio em janelas de 25ms de duração e salto de 10ms (*overlap* de 40%). Em seguida são calculados os MFCCs, com 40 bandas *mel* no banco de filtros. Estes coeficientes são utilizados como entrada da rede LSTM.

3.3 Rede Pré-treinada

Neste trabalho optou-se em utilizar um modelo pré-treinado de uma rede neural LSTM, ao invés de efetuar todo o treinamento. O modelo pode ser encontrado em [30], sendo possível baixar tanto o código fonte utilizado para treinar a rede, quanto o arquivo com o modelo salvo. O

Figura 3.1: Diagrama de Blocos do Sistema de *Speaker Diarization* Implementado



Fonte: do Autor.

treinamento da rede teve como base [64], dos mesmos autores de [66], utilizando a mesma rede na implementação.

O treinamento da rede é feito utilizando três bases de dados¹ diferentes: VCTK [11], LibriSpeech [40] e VoxCeleb [12]. Antes da etapa de segmentação os sinais de áudio passam por um VAD baseado apenas em energia, diferentemente do artigo base [64] que usa um modelo GMM para efetuar a tarefa.

Estrutura da Rede

O número de exemplos a serem treinados é chamado de *batch-size*, usualmente representados por uma matriz. Cria-se uma matriz de exemplos com dimensões $N \times M$, sendo N o número de diferentes locutores, e

¹Uma breve descrição das bases utilizadas pode ser encontrada no Apêndice C

M o número de sentenças de cada locutor. A matriz x_{ji} ($1 \leq j \leq N$ e $1 \leq i \leq M$) representa os parâmetros MFCCs da sentença i pertencente ao locutor j . Aplica-se a matriz \mathbf{X}_{ji} como entrada da rede LSTM, com estrutura apresentada na Tabela 3.1, com taxa de aprendizagem (*learning rate*) de 0.001.

Tabela 3.1: Estrutura da Rede LSTM.

Número de Camadas Ocultas	3
Dimensão das Camadas Ocultas	768
Dimensão de Projeção	256
Batch Size (N x M)	64×10

Fonte: do Autor.

Rede Pré-treinada como Extrator de Parâmetros

A rede foi treinada para aplicações de *Speaker Verification*, em um cenário TI-SV (*Text Independent Speaker Verification*), dessa forma os parâmetros extraídos da rede funcionam para qualquer sinal de fala arbitrário. A motivação ao usar uma rede para *Speaker Verification*, na implementação de um sistema *Speaker Diarization*, deve-se ao fato de que a rede aprende a extrair as informações mais relevantes na fala de um locutor. Sendo assim, a rede torna-se uma boa fonte para extração de parâmetros de comparação entre locutores. Os vetores de saída da função de ativação da última camada oculta da rede LSTM treinada são conhecidos como *d-vectors*². Utiliza-se portanto como parâmetro final a norma L2 dos *d-vectors* dada por

$$d_{ji} = \frac{\vec{d}}{\|\vec{d}\|^2}, \quad (3.1)$$

possuindo como dimensões $N \times 256$, onde N representa o número total de locutores, e 256 é a dimensão de projeção da rede LSTM, ou seja, para cada sentença de um dado locutor é gerado um vetor com 256 coeficientes. Uma das vantagens dessa representação é, independente do tamanho do sinal de áudio na entrada da rede, o *d-vector* desse

²Conceito apresentado no Apêndice B

senal sempre será de dimensão 1×256 . Estes vetores serão utilizados na etapa de *clustering* para determinar o número de locutores, como também a qual locutor cada trecho pertence.

3.4 Etapa de *Clustering*

Os algoritmos de *Spectral Clustering* e *K-means* foram utilizados respectivamente para distinguir os locutores a partir dos *d-vectors*, e determinar o número de locutores presentes no canal, como também atribuir uma marcação a cada segmento com seu devido locutor. Ambos algoritmos são revisados e explicados no Apêndice B.

Spectral Clustering

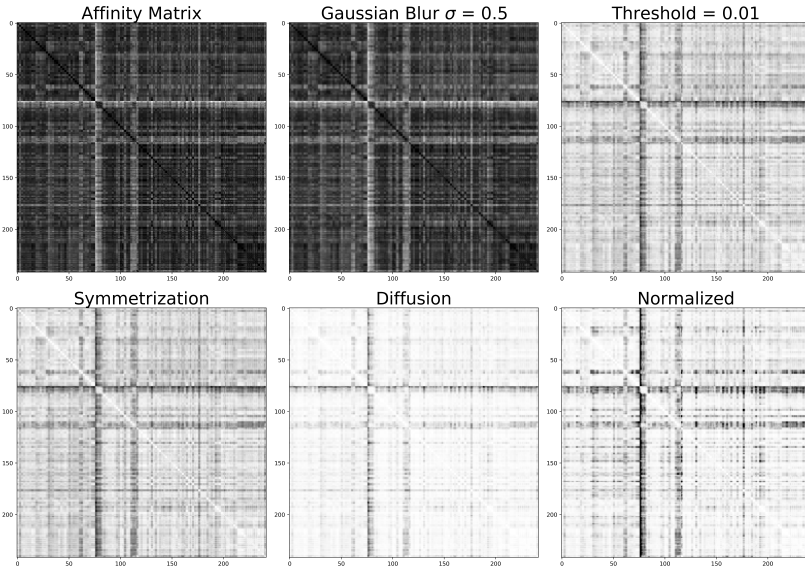
A partir dos *d-vectors* extraídos pela rede, aplica-se o algoritmo *Spectral Clustering*, que consiste na construção da matriz de similaridade do sinal de fala. Logo após, uma série de processos de refinamento são aplicados na matriz. Estes refinamentos são baseados na localidade temporal do sinal de fala – segmentos de fala contínuos possuem parâmetros muito próximos, valores estes perceptíveis ao utilizar a matriz de similaridade, onde os parâmetros com maior similaridade possuem valor próximo a zero, e alocados na diagonal principal da matriz.

Os processos de refinamento da matriz são feitos com o uso da biblioteca SciPy, aplicando-se funções específicas de processamento de imagens. A Figura 3.2 apresenta uma representação gráfica dos efeitos de refinamento, gerados sobre o arquivo `speech-librivox-0012.wav` da base de dados MUSAN. Os passos de refinamento são descritos pelos seguintes passos:

- (i) *Gaussian Blur* para suavizar as bordas, com variância de 0.5;
- (ii) *Thresholding*, zera os elementos com valores absolutos inferiores a 0.001;
- (iii) Simetrização, torna a matriz simétrica em relação a sua diagonal principal;
- (iv) Difusão, afina as bordas entre as seções da matriz de similaridade, com o intuito de melhor distinguir os locutores;

- (v) *Row-wise Max Normalization*, passo final que reescala a matriz para eliminar efeitos indesejáveis de escala.

Figura 3.2: Representação gráfica dos passos de refinamento na matriz de similaridades.



Fonte: do Autor.

Após os passos de refinamento, aplica-se a decomposição em valores singulares sobre a matriz refinada. Dado um número n de auto-valores ($\lambda_1 > \lambda_2 > \dots > \lambda_n$), o número de *clusters* \tilde{k} é determinado por

$$\tilde{k} = \arg \max_{1 \leq k \leq n} \frac{\lambda_k}{\lambda_{k+1}}. \quad (3.2)$$

K-means

Sejam os auto-vetores correspondentes aos maiores auto-valores \tilde{k} , definidos como $v_1, v_2, \dots, v_{\tilde{k}}$. Utiliza-se uma variação do algoritmo *K-means*, conhecida como *K-means++* definida como

$$k = \arg \max_{k \geq 1} \text{MSCD}'(\tilde{k}), \quad (3.3)$$

onde o MSCD (*Means Squared Cosine Distances*) é dado por

$$\text{MSCD} = \frac{A \cdot B}{\|A\| \|B\|}, \quad (3.4)$$

onde A e B correspondem a dois diferentes segmentos clusterizados pelo *Spectral Clustering*. Para determinar o número de locutores k , utiliza-se o “*elbow*” (função custo J) das derivadas do MSCD entre cada parâmetro. Ao mapear os parâmetros para um centróide, a partir do “*elbow*”, é possível produzir as marcações de locutor para cada trecho de fala. Para cada locutor é atribuído um número identificador. Para o caso deste trabalho, onde os cenários possuem apenas dois locutores, as marcações são feitas utilizando os números 0 e 1 como identificadores.

Um exemplo gráfico da saída resultante do sistema de *Speaker Diarization* é ilustrada na Figura 3.3. O arquivo utilizado neste exemplo foi gerado sinteticamente, a partir da concatenação entre dois sinais de fala de locutores distintos. Na imagem duas marcações são ilustradas, uma de referência com a informação real dos trechos de fala de cada locutor, a outra uma hipótese gerada pelo sistema. Ainda na Figura 3.3 encontra-se uma janela aproximada de um pequeno trecho marcado, nota-se que a hipótese consta mudanças de locutor em pequenos intervalos, apresentando uma certa taxa de erro, avaliada posteriormente.

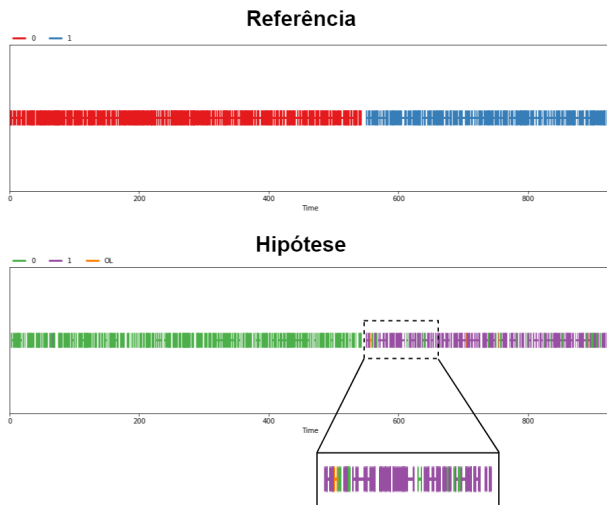
3.5 Avaliação do Sistema

A métrica padrão estabelecida pelo NIST, para avaliação de um sistema de *Speaker Diarization*, assim como a comparação com outros sistemas, é chamada de *Diarization Error Rate* (DER). A métrica é definida como

$$\text{DER} = \frac{\text{False Alarm} + \text{Missed Detection} + \text{Confusion}}{\text{total}}, \quad (3.5)$$

onde *False Alarm* (FA) corresponde a duração de trechos sem fala classificados como fala; *Missed Detection* (Miss) é a duração de trechos de fala classificados como sem fala; e por fim *Confusion* (Con) é a duração de trechos classificados com locutor errado. Com a soma de todas as durações de possíveis erros sobre a duração total do sinal, obtêm-se uma taxa percentual de erro.

Figura 3.3: Marcações de Referência e Hipótese gerada pelo sistema *Speaker Diarization* implementado.



Fonte: do Autor.

Todas métricas citadas são aplicadas sobre marcações como as da Figura 3.3; ambas métricas e marcações foram efetuadas em linguagem PYTHON, com o uso da biblioteca Pyannote [7]. Essa biblioteca *open-source* foi especialmente desenvolvida para trabalhos de pesquisa em *Speaker Diarization*.

3.6 Cenários de Avaliação

Para avaliar o desempenho e algumas características do sistema de *Speaker Diarization* implementado, foram realizados experimento em dois cenários diferentes. Em tais experimentos buscou-se a avaliação de generalidade do sistema, para isso uma base de dados não conhecida pela rede neural foi utilizada, a base MUSAN [57]. Para ambos cenários há presença apenas de dois locutores em um mesmo canal de áudio.

No primeiro cenário montado utilizou-se sinais de fala independente da língua falada ou nacionalidade do locutor. A base MUSAN conta com 17 línguas diferentes faladas. A Tabela 3.2 apresenta todos os 17

idiomas. Já o segundo cenário, apenas locutores com sinais de fala na língua inglesa foram utilizados.

Tabela 3.2: Lista com os 17 idiomas presentes na base MUSAN.

Línguas	<i>arabic</i>	<i>chinese</i>	<i>danish</i>	<i>dutch</i>	<i>english</i>	<i>french</i>
	german	hebrew	hungarian	italian	japanese	latin
	<i>polish</i>	<i>portuguese</i>	<i>russian</i>	<i>spanish</i>	<i>tagalog</i>	

3.7 Resultados e Discussões

Os resultados obtidos ao avaliar o sistema para o Cenário 1 são informados na Tabela 3.3, constando os resultados dos percentuais mínimo, médio e máximo das componentes individuais e o total da taxa de erro DER. Como parâmetro de comparação, também são apresentados os resultados obtidos pelo Google no artigo “*Speaker Diarization with LSTM*”, em um cenário com sinais de fala de dois locutores, conversando em diferentes idiomas, em uma linha telefônica.

Tabela 3.3: Cenário com locutores de países distintos

Método	Confusion (%)		FA (%)		Miss Detection (%)		Total DER (%)	
Implementado	médio	6.75±1.38	médio	11.06±0.51	médio	9.96±4.59	médio	25.77±7.82
	mínimo	0.00	mínimo	0.00	mínimo	4.44	mínimo	6.88
	máximo	45.65	máximo	39.31	máximo	99.02	máximo	132.09
Artigo Base (Valores Médios)	12.0		2.2		4.6		18.8	

Fonte: do Autor.

As mesmas métricas foram aplicadas no Cenário 2, conversas entre dois locutores em língua inglesa. Os valores mínimos, médios e máximos das taxas de *Confusion*, *False Alarm* e *Missed Detection*, como também a DER total são apresentadas na Tabela 3.4. Os resultados obtidos pelo Google, para um cenário similar ao descrito, também são apresentados.

Algumas características podem ser observadas ao agrupar todas métricas, de ambos cenários, a um histograma. As Figuras 3.4 e 3.5 apresentam respectivamente, os histogramas das métricas extraídas dos Cenários 1 e 2.

Os valores de *Miss Detection* e *False Alarm* comumente representam falhas provenientes do *Voice Activity Detection* (VAD). A influência do

Tabela 3.4: Cenário com dois locutores Americanos.

Método	Confusion (%)		FA (%)		Miss Detection (%)		Total DER (%)	
Implementado	médio	3.07 ± 1.08	médio	9.09 ± 0.16	médio	4.44±0.13	médio	17.27 ± 1.51
	mínimo	0.00	mínimo	12.81	mínimo	5.62	mínimo	3.48
	máximo	68.03	máximo	21.25	máximo	23.32	máximo	82.20
Artigo Base (Valores Médios)	5.97		2.51		4.06		12.54	

Fonte: do Autor.

False Alarm na taxa total DER é alta comparada com o artigo base, efeito este devido ao VAD utilizado ser para aplicações de *VoIP*. Isso ocorre devido o algoritmo passar sinais de ruído como fala, ocorrência comum em sistemas de *VoIP*. A taxa de *Miss Detection*, em contraste ao *False Alarm*, apresenta valor muito próximo ao artigo base.

Um dos dados mais interessantes de se observar é a taxa de Confusão, que para ambos cenários representa uma pequena parcela da DER. Essa informação demonstra que o sistema consegue distinguir bem as diferenças entre locutores, ou seja, tanto a etapa de extração de parâmetros via rede LSTM, quanto a de *clustering* apresentam bons desempenhos.

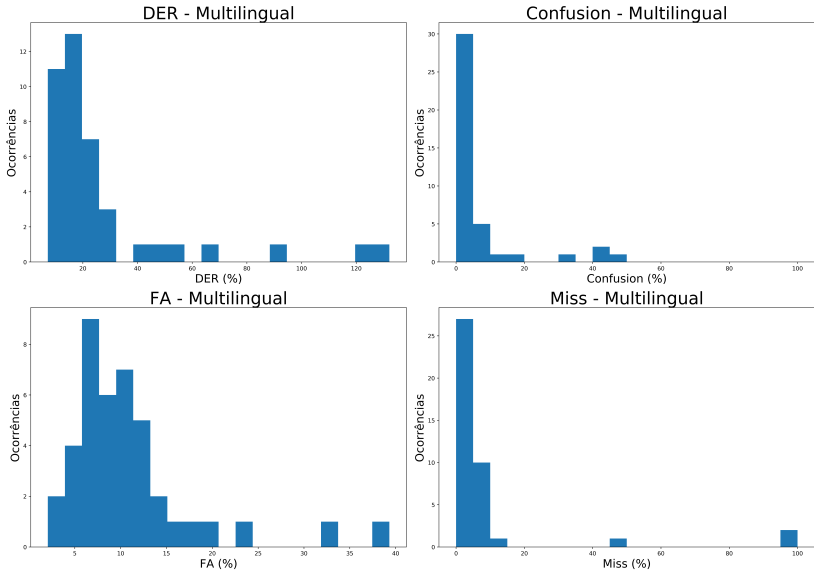
3.7.1 Comparativo entre Sistema Base e Implementado

Comparado ao sistema base apresentado em [66], o sistema de *Speaker Diarization* implementado possui algumas deficiências. A primeira delas encontra-se na etapa de pré-processamento do sistema, onde o VAD utilizado é focado em aplicações de telefonia *VoIP*, enquanto em [66] usa-se um modelo GMM para a tarefa, especificamente para aplicações de *Speaker Verification*. Este déficit é observado ao comparar os valores de *False Alarm*, entre o sistema base e o implementado, nas Tabelas 3.3 e 3.4. A taxa é quase 4 vezes maior no sistema implementado, demonstrando um grande gargalo na etapa que usa o VAD.

3.8 Tendências Futuras

Uma das tendências em pesquisas de *Speaker Diarization* é a substituição da etapa de *clustering*, tendo como ideia a substituição por um sistema baseado puramente em redes neurais recorrentes (RNNs). Esse tipo de sistema é chamado de “*Fully Supervised Speaker Diarization*” [69].

Figura 3.4: Histogramas referentes às métricas do sistema no Cenário 1.



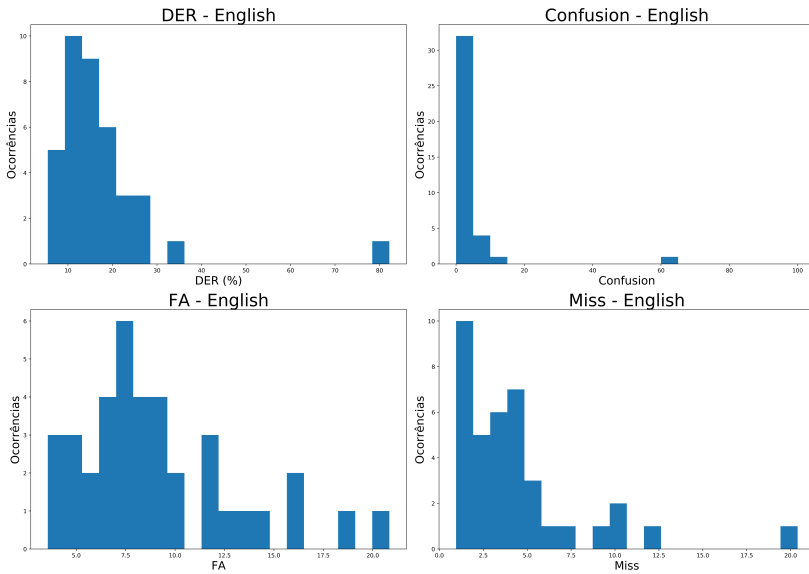
Fonte: do Autor.

A ideia por traz dessa nova metodologia é modelar cada locutor por uma instância de uma RNN, onde estas instâncias compartilham dos mesmo parâmetros. Um número ilimitado de instâncias são geradas, onde os estados da RNN presentes em cada instância corresponde aos diferentes locutores, sendo assim a identificação destes locutores em diferentes instantes de tempo, é feita automaticamente.

Outra tendência é o uso de diferentes topologias e arquiteturas de redes neurais, para efetuar a extração dos parâmetros d -vectors. Um exemplo é apresentado em [68], onde utiliza-se uma topologia de rede baseada na ResNet [22], usada em aplicações de reconhecimento de padrões em imagens. Pode-se fazer uso dessas redes ao tratar, por exemplo, os MFCCs no tempo como uma imagem, gerando assim como saída uma nova representação de d -vectors. Os passos posteriores nestas tendências são similares ao implementado neste trabalho, utilizando algoritmos de *clustering* como, por exemplo, o AHC, *Spectral Clustering* e *K-means++*.

Novos parâmetros de extração estão sendo utilizados também, estes

Figura 3.5: Histogramas referentes às métricas do sistema no Cenário 2.



Fonte: do Autor.

parâmetros são conhecidos como *x-vectors* [53]. Pouco se sabe sobre estes novos vetores de representação, porém eles são obtidos de forma muito similar aos *d-vectors*, retira-se os vetores antes da camada *Fully Connected* de uma nova topologia de rede neural profunda chamada de *Time-Delay Neural Network* (TDNN). Atualmente os *x-vectors* são extraídos utilizando o *ToolKit* para processamento de fala Kaldi [45].

CAPÍTULO 4

Conclusão

Neste trabalho, foi apresentado um sistema de *Speaker Diarization* baseado em técnicas de aprendizagem profunda e *clustering*. Foi apresentada uma breve revisão do estado da arte, contendo a evolução dos sistemas, características das bases de dados usadas nessas aplicações, arquiteturas de modelos de locutores e tipos de extração de parâmetros. No desenvolvimento do sistema foi escolhido uma base de dados não utilizada no treinamento da rede neural, dessa forma sendo avaliada a generalização da rede ao lidar com locutores desconhecidos por ela. Seguiu-se a metodologia considerada o estado da arte, e foram comparados os resultados obtidos do sistema implementado com o de referência, comparação feita em dois cenários. A acurácia do sistema implementado atingiu, respectivamente, nos Cenários 1 e 2 valores de 25.77% e 17.27%, contra os 18.8% e 12.54% do sistema base. Foram feitas discussões sobre possíveis fatores que influenciaram essa discrepância de acurácia. Um dos principais motivos deve-se ao modelo do VAD utilizado, tem como foco aplicações de telefonia *VoIP*, que tende a aumentar a taxa de *False Alarm*, já no artigo base o VAD tem foco exclusivo em aplicações de *Speaker Verification*, sendo mais robusto. Outra desvantagem encontra-se no modelo da rede LSTM utilizada,

sendo esta treinada com cerca de 2400 horas de áudio, contra as 18 800 horas utilizados no treinamento da rede do sistema base.

Trabalhos futuros

A fim de melhorar o desempenho do sistema implementado, outras técnicas de implementação de um VAD devem ser utilizadas, a fim de baixar a taxa de erro de *Miss Detection*. Outra possibilidade seria o uso de outro tipo de rede neural para extração dos *d-vectors*, tais como a GRU (*Gated Recurrent Unit*) [69] e ResNet (*Residual Network*) [28]. Para a etapa de *clustering* outras técnicas para separação podem ser investigadas, como o uso de redes neurais para efetuar a tarefa. Algumas técnicas utilizando redes BiLSTM (*Bidimensional Long-term Short Memory*) [23] para o uso de separação de locutores em sinais de fala sobrepostas, tendo grande potencial no uso para sistemas de *Speaker Diarization*.

Referências Bibliográficas

- [1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.
- [2] Jitendra Ajmera and Chuck Wooters. A robust speaker clustering algorithm. In *2003 IEEE Workshop on Automatic Speech Recognition and Understanding (IEEE Cat. No. 03EX721)*, pages 411–416. IEEE, 2003.
- [3] Xavier Anguera, Simon Bozonnet, Nicholas Evans, Corinne Fre-douille, Gerald Friedland, and Oriol Vinyals. Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech & Language Processing*, 20:356–370, 02 2012.
- [4] Joyanta Basu, Soma Khan, Rajib Roy, Madhab Pal, Tulika Basu, Milton Samirakshma Bepari, and Tapan Kumar Basu. An overview of speaker diarization: Approaches, resources and challenges. In *2016 Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Data-*

- bases and Assessment Techniques (O-COCOSDA)*, pages 166–171. IEEE, 2016.
- [5] Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.
- [6] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.
- [7] Hervé Bredin. pyannote. metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems. In *INTERSPEECH*, pages 3587–3591, 2017.
- [8] Hervé Bredin. Tristounet: Triplet loss for speaker turn embedding. *ICASSP 2017*, apr 2017.
- [9] Yu-hsin Chen, Ignacio Lopez-Moreno, Tara N Sainath, Mirkó Vissontai, Raziq Alvarez, and Carolina Parada. Locally-connected and convolutional neural networks for small footprint speaker recognition. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.
- [10] Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjali Kannan, Ron J. Weiss, Kanishka Rao, Katya Gonina, Navdeep Jaitly, Bo Li, Jan Chorowski, and Michiel Bacchiani. State-of-the-art speech recognition with sequence-to-sequence models. *CoRR*, abs/1712.01769, 2017.
- [11] Veaux Christophe, Yarnagishi Junichi, and MacDonald Kirsten. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. *The Centre for Speech Technology Research (CSTR)*, 2016.
- [12] Joon Son Chung, Arsha Nagrani, and Andrew Senior. Voxceleb2: Deep speaker recognition. *arXiv preprint arXiv:1806.05622*, 2018.
- [13] Stephen Cook. Speech recognition howto. *The Linux Documentation Project*, 2002.

- [14] Najim Dehak, Patrick J. Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2011.
- [15] Nicholas Evans, Simon Bozonnet, Dong Wang, Corinne Fredouille, and Raphaël Troncy. A comparative study of bottom-up and top-down approaches to speaker diarization. *IEEE Transactions on Audio, speech, and language processing*, 20(2):382–392, 2012.
- [16] Gerald Friedland, Oriol Vinyals, Yan Huang, and Christian Muller. Prosodic and other long-term features for speaker diarization. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(5):985–993, 2009.
- [17] Olivier Galibert and Juliette Kahn. The first official repere evaluation. In *First Workshop on Speech, Language and Audio in Multimedia*, 2013.
- [18] Zhenhao Ge, Ananth N. Yer, Srinath Cheluvvaraja, and Aravind Ganapathiraju. Speaker change detection using features through a neural network speaker classifier. *2017 Intelligent Systems Conference (IntelliSys)*, sep 2017.
- [19] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU press, 2012.
- [20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [21] Simon S Haykin et al. *Neural networks and learning machines/Simon Haykin*. New York: Prentice Hall,, 2009.
- [22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [23] John R Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe. Deep clustering: Discriminative embeddings for segmentation and separation. In *2016 IEEE International Conference on*

- Acoustics, Speech and Signal Processing (ICASSP)*, pages 31–35. IEEE, 2016.
- [24] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, aug 1997.
- [25] Jesper Hojvang Jensen, Daniel PW Ellis, Mads G Christensen, and Soren Holdt Jensen. Evaluation distance measures between gaussian mixture models of mfccs. 2007.
- [26] Aishwary Joshi, Mohit Kumar, and Pradip K Das. Speaker diarization: A review. In *2016 International Conference on Signal Processing and Communication (ICSC)*, pages 191–196. IEEE, 2016.
- [27] Patrick Kenny, Douglas Reynolds, and Fabio Castaldo. Diarization of telephone conversations using factor analysis. *IEEE Journal of Selected Topics in Signal Processing*, 4(6):1059–1070, 2010.
- [28] Chao Li, Xiaokong Ma, Bing Jiang, Xiangang Li, Xuewei Zhang, Xiao Liu, Ying Cao, Ajay Kannan, and Zhenyao Zhu. Deep speaker: an end-to-end neural speaker embedding system. *arXiv preprint arXiv:1705.02304*, 2017.
- [29] Lantian Li, Dong Wang, Zhiyong Zhang, and Thomas Fang Zheng. Deep speaker vectors for semi text-independent speaker verification. *arXiv preprint arXiv:1505.06427*, 2015.
- [30] Dalon Lobo. *Modelo Pré-treinado de uma Rede LSTM para Speaker Verification*. 2019. <https://github.com/dalonlobo/diarization-experiments/tree/master/models>.
- [31] Dimitris G Manolakis and Vinay K Ingle. *Applied digital signal processing: theory and practice*. Cambridge University Press, 2011.
- [32] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, pages 18–25, 2015.
- [33] Sylvain Meignier and Teva Merlin. Lium spkdiarization: an open source toolkit for diarization. In *CMU SPUD Workshop*, 2010.

- [34] Tom M Mitchell. Does machine learning really work? *AI magazine*, 18(3):11–11, 1997.
- [35] Daniel Moraru, Sylvain Meignier, Corinne Fredouille, Laurent Besacier, and Jean-François Bonastre. The elisa consortium approaches in broadcast news speaker segmentation during the nist 2003 rich transcription evaluation. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–373. IEEE, 2004.
- [36] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. Voxceleb: a large-scale speaker identification dataset. *arXiv preprint arXiv:1706.08612*, 2017.
- [37] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.
- [38] Athanasios Noulas, Gwenn Englebienne, and Ben JA Krose. Multimodal speaker diarization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):79–93, 2011.
- [39] Christopher Olah. *Understanding LSTM networks*. 2015. <http://colah.github.io/posts/2015-08-Understanding-LSTMs>.
- [40] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE, 2015.
- [41] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.
- [42] Gay Paul, Khoury Elie, Meignier Sylvain, Odobez Jean-Marc, and Deleglise Paul. A conditional random field approach for audio-visual people diarization. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 116–120. IEEE, 2014.

- [43] Ms. Monali R. Pimpale, Prof. Shanthi Therese, and Prof. Vinayak Shinde. A survey on: Sound source separation methods. *International Journal of Computer Engineering in Research Trends (IJ-CERT)*, 3(11):580–584, November 2016.
- [44] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *Advances in neural information processing systems*, pages 3360–3368, 2016.
- [45] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldı speech recognition toolkit. Technical report, IEEE Signal Processing Society, 2011.
- [46] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2847–2854. JMLR. org, 2017.
- [47] D.A Reynolds and Lincoln Lab. Approaches and applications of audio diarization. *Proceedings. (ICASSP '05). IEEE International Conference on*, vol.(5):v/949–v/952, mar 2005.
- [48] Douglas A Reynolds, Thomas F Quatieri, and Robert B Dunn. Speaker verification using adapted gaussian mixture models. *Digital signal processing*, 10(1-3):19–41, 2000.
- [49] Douglas A Reynolds and P Torres-Carrasquillo. The mit lincoln laboratory rt-04f diarization systems: Applications to broadcast audio and telephone conversations. Technical report, MASSACHUSETTS INST OF TECH LEXINGTON LINCOLN LAB, 2004.
- [50] Jamal E Rougui, Mohammed Rziza, Driss Aboutajdine, Marc Gellon, and José Martinez. Fast incremental clustering of gaussian mixture speaker models for scaling up retrieval in on-line broadcast. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 5, pages V–V. IEEE, 2006.

- [51] Neville Ryant, Mark Liberman, and Jiahong Yuan. Speech activity detection on youtube using deep neural networks. In *INTERSPEECH*, pages 728–731. Lyon, France, 2013.
- [52] Sergey Salishev, Andrey Barabanov, Daniil Kocharov, Pavel Skrelin, and Mikhail Moiseev. Voice activity detector (vad) based on long-term mel frequency band features. In *International Conference on Text, Speech, and Dialogue*, pages 352–358. Springer, 2016.
- [53] Gregory Sell, David Snyder, Alan McCree, Daniel Garcia-Romero, Jesús Villalba, Matthew Maciejewski, Vimal Manohar, Najim Dehak, Daniel Povey, Shinji Watanabe, et al. Diarization is hard: Some experiences and lessons learned for the jhu team in the inaugural dihard challenge. In *Proc. Interspeech*, pages 2808–2812, 2018.
- [54] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, R. J. Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgianakis, and Yonghui Wu. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. *CoRR*, abs/1712.05884, 2017.
- [55] Elizabeth Shriberg. Higher-level features in speaker recognition. In *Speaker Classification I*, pages 241–259. Springer, 2007.
- [56] Matthew A Siegler, Uday Jain, Bhiksha Raj, and Richard M Stern. Automatic segmentation, classification and clustering of broadcast news audio. In *Proc. DARPA speech recognition workshop*, volume 1997, 1997.
- [57] David Snyder, Guoguo Chen, and Daniel Povey. Musan: A music, speech, and noise corpus. *arXiv preprint arXiv:1510.08484*, 2015.
- [58] Stanley Smith Stevens, John Volkman, and Edwin B Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.
- [59] Vibha Tiwari. Mfcc and its applications in speaker recognition. *International journal on emerging technologies*, 1(1):19–22, 2010.

- [60] Jerry Tobias. *Foundations of modern auditory theory*. Elsevier, 2012.
- [61] Sue E Tranter and Douglas A Reynolds. An overview of automatic speaker diarization systems. *IEEE Transactions on audio, speech, and language processing*, 14(5):1557–1565, 2006.
- [62] Ehsan Variiani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. Deep neural networks for small footprint text-dependent speaker verification. *IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, pages 4080–4084., 2014.
- [63] Arlindo Veiga, Carla Lopes, and Fernando Perdigão. Speaker diarization using gaussian mixture turns and segment matching. *Proc. FALA*, 2010.
- [64] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4879–4883. IEEE, 2018.
- [65] Chong Wang. Accurate online speaker diarization with supervised learning, nov 2018.
- [66] Quan Wang, Carlton Downey, Li Wan, Philip Andrew Mansfield, and Ignacio Lopz Moreno. Speaker diarization with lstm. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5239–5243. IEEE, 2018.
- [67] Ruiqing Yin, Hervé Bredin, and Claude Barras. Speaker change detection in broadcast tv using bidirectional long short-term memory networks. In *Interspeech 2017*. ISCA, 2017.
- [68] Takuya Yoshioka, Zhuo Chen, Dimitrios Dimitriadis, William Hinthorn, Xuedong Huang, Andreas Stolcke, and Michael Zeng. Meeting transcription using virtual microphone arrays. *arXiv preprint arXiv:1905.02545*, 2019.
- [69] Aonan Zhang, Quan Wang, Zhenyao Zhu, John Paisley, and Chong Wang. Fully supervised speaker diarization. In *ICASSP 2019-2019*

IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6301–6305. IEEE, 2019.

- [70] Inbar Mosseri and Oran Lang. Looking to Listen: Audio-Visual Speech Separation. <https://ai.googleblog.com/2018/04/looking-to-listen-audio-visual-speech.html>. Acesso em: 06 maio 2019.
- [71] Katherine Chou, Product Manager and Chung-Cheng Chi. Understanding Medical Conversations. <https://ai.googleblog.com/2017/11/understanding-medical-conversations.html>. Acesso em: 06 maio 2019.

Conceitos Básicos de Processamento Digital de Sinais

Sinais de fala são intrinsecamente um fenômeno analógico, representado por uma onda mecânica propagando por um meio. A gravação desses sinais mecânicos, utilizando um microfone, converte a onda mecânica em um sinal elétrico e contínuo. Esse sinal é filtrado e convertido em um sinal discreto por um conversor analógico-digital (A/D), com certa taxa de amostragem e nível de quantização.

A fala possui uma largura de banda baixa entre 50Hz a 3.4kHz, sendo assim uma taxa de amostragem de 8kHz seria suficiente para representá-la. Porém, alguns sistemas de processamento de fala utilizam taxas de amostragem de 16kHz, por exemplo um ASR (*Automatic Speech Recognition*) [13], pois a essa taxa obtêm-se maior fidelidade em altas frequências.

A diferenciação de locutores utilizando apenas sua representação discreta no tempo é uma tarefa árdua, se não beira o impossível. Para isso representações no domínio da frequência são mais usuais como modo de extração de parâmetros para a diferenciação.

Os sinais de fala podem facilmente alcançar dimensões espantosas, por exemplo, 30 segundos de uma gravação de fala com taxa de amos-

tagem de 16kHz possui cerca de meio milhão de amostras. Dessa forma no escopo de *machine learning* e *deep learning* é de suma importância que seja efetuada a redução de dimensionalidade dos sinais, por se tratar de algoritmos complexos.

Algumas técnicas para melhoria de desempenho, redução de dimensionalidade e de extração de parâmetros serão tratadas nesta seção.

A.1 Transformada de Fourier e Espectrograma

A análise de sinais de áudio em geral torna-se mais simples no domínio da frequência, obtido a partir da Transformada de Fourier. A transformada de Fourier converte a informação de tempo ou espaço em componentes de magnitude e fase, obtendo-se representações em cada frequência ou um espectro de frequências do sinal.

Seja um sinal discreto $x[n]$ de comprimento finito N , a sua transformada discreta de Fourier (DFT) [31] pode ser definida como:

$$X[k] = \sum_{k=0}^{N-1} x[k]e^{-j(\frac{2\pi n}{N})k}. \quad (\text{A.1})$$

O sinal $X[k]$ é a representação no domínio da frequência do sinal $x[n]$, com duração finita N e com frequências definidas como $\omega_k = 2\pi k/N$, $0 \leq k \leq N - 1$. Portanto ao aumentar o número de amostras N , a resolução do sinal $X[k]$ será maior, levando a um espectro de frequência mais detalhado.

O conteúdo de frequência de sinais de fala apresenta variações no tempo, ou seja, é um sinal não-estacionário. A DFT apresenta alguns problemas ao lidar com a análise desses sinais, pois utiliza o sinal inteiro nas operações o que gera aumento da complexidade computacional e dimensionalidade do problema. Portanto, a DFT é mais apropriada para análise de sinais estacionários, onde o conteúdo de frequência não muda com o tempo.

Para contornar este problema representa-se o sinal de fala em trechos de curta duração (*frames*), pois o fazendo considera-se que o sinal de cada trecho é aproximadamente estacionário. A representação na frequência para esses casos é a transformada de Fourier para tempos

curtos (STFT) [31], definida por:

$$X[n, k] \triangleq \sum_{m=0}^{L-1} w[m]x[nH + m]e^{-j(\frac{2\pi n}{N})k}m, \quad (\text{A.2})$$

sendo L o tamanho da função janela $w[m]$, o tamanho do salto entre *frames* (H) usualmente é menor que o tamanho da janela L , isso introduz sobreposições (*overlap*) entre os *frames* adjacentes. Além disso, a função janela $w[n]$ é utilizada para redução de efeitos de vazamento espectral, portanto é utilizada afim de garantir a continuidade nas bordas dos *frames*. A função janela escolhida foi a de *Hanning*, definida como

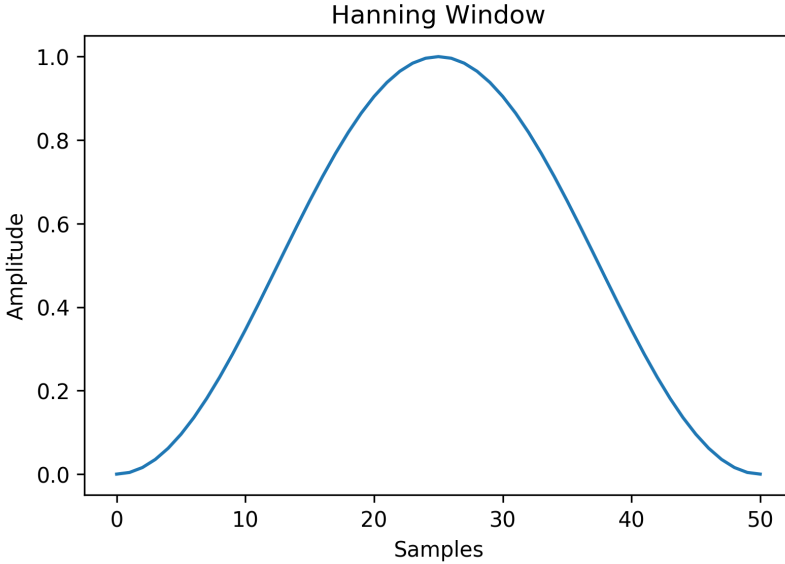
$$w[n] = 0.5 - 0.5 \cos(2\pi n/M), \quad 0 \leq n \leq N, \quad (\text{A.3})$$

e ilustrada na Figura A.1.

Entretanto, o janelamento tende a zerar as amostras próximas às fronteiras do *frame*. Devido a isso, as informações dessas amostras não são levadas em consideração ao efetuar a STFT. Para minimizar esse efeito utiliza-se a técnica de *Overlap-Add* abordada na Seção A.2.

Um exemplo do efeito da técnica de *overlap* é mostrado na Figura A.2, sendo utilizado como sinal uma onda senoidal segmentada em 3 *frames*. A Figura A.2a apresenta o efeito do janelamento sem o uso de sobreposições entre os *frames*. Percebe-se a perda de informação da senoide nas extremidades destes *frames*, demonstrando de forma visual o um problema ao usar janelamento. A Figura A.2b mostra o efeito de sobreposição sobre as 3 janelas de Hanning, com sobreposição de 50% a soma resultante se aproxima de uma constante. Ao multiplicar essa janela resultante pela senoide é obtido o sinal apresentado na Figura A.2c. É perceptível a importância do uso do *overlap*, demonstrando a pouca perda de informação do sinal em contraponto ao caso sem *overlap*.

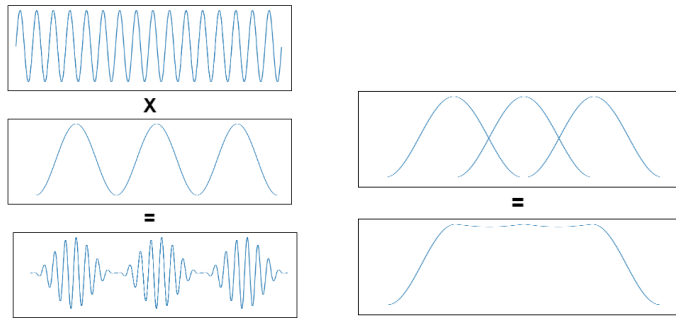
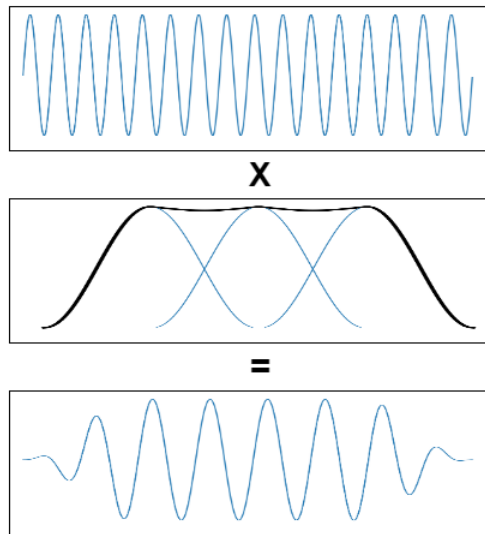
Figura A.1: Janela de Hanning com tamanho 51 amostras.



Fonte: do Autor.

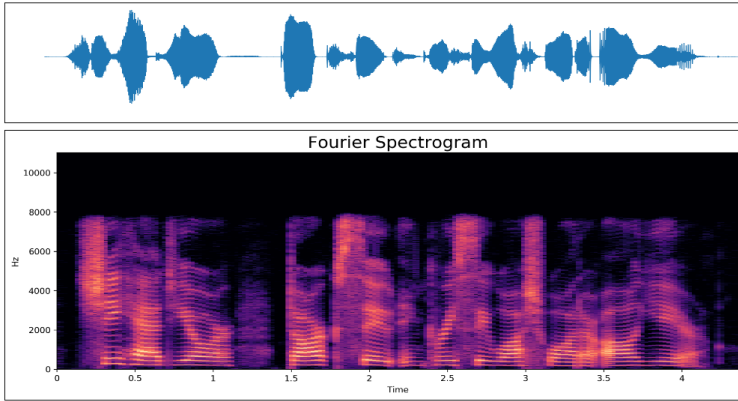
A magnitude de $X[n, k]$, também é chamada de espectrograma, que nada mais é que as variações do espectro de um sinal, sendo uma poderosa ferramenta de análise, seja computacional ou visual. O espectrograma pode ser tratado como uma imagem 3D, possuindo um canal de cor, tempo e frequência, viabilizando análises mais refinadas com técnicas de processamento digital de imagens.

A Figura A.3 ilustra a representação visual de um espectrograma de um sinal de fala com 1 minuto e 25 segundos de duração, representado pela sua forma temporal em azul. Abaixo do sinal de fala encontra-se o seu respectivo espectrograma, contendo as componentes de frequências em cada instante de tempo ($1/f_s$), espectrograma este calculado com taxa de *overlap* de 50% e tamanho de janelas de 25ms. As componentes de frequência com maior energia apresentam cores mais claras, já as cores mais escuras representam componentes com energia baixa ou até mesmo nula.

Figura A.2: Efeitos do *Overlap* sobre sinais janelados.(a) Sinal janelado sem *Overlap*.(b) Janela resultante do *Overlap*.(c) Sinal janelado com *Overlap*.

Fonte: do Autor.

Figura A.3: Sinal de fala e sua representação em espectrograma.



Fonte: do Autor.

A.2 Método *Overlap-Add*

Em processamento de sinais o método *overlap-add* é utilizado como uma forma eficiente de se efetuar uma convolução discreta de seqüências muito longas, sendo a peça fundamental do funcionamento da STFT apresentada na Seção A.1.

Seja $w[n]$ uma seqüência de tamanho M e $x[n]$ uma seqüência de comprimento ilimitado, a convolução linear entre as duas seqüências é:

$$y[n] = \sum_{l=0}^{M-1} w[l]x[n-l], \quad (\text{A.4})$$

segmenta-se $x[n]$ em um conjunto de seqüências contíguas de N amostras [31], como:

$$x[n] = \sum_{m=0}^{\infty} x_m[n - mN] \quad (\text{A.5})$$

em que

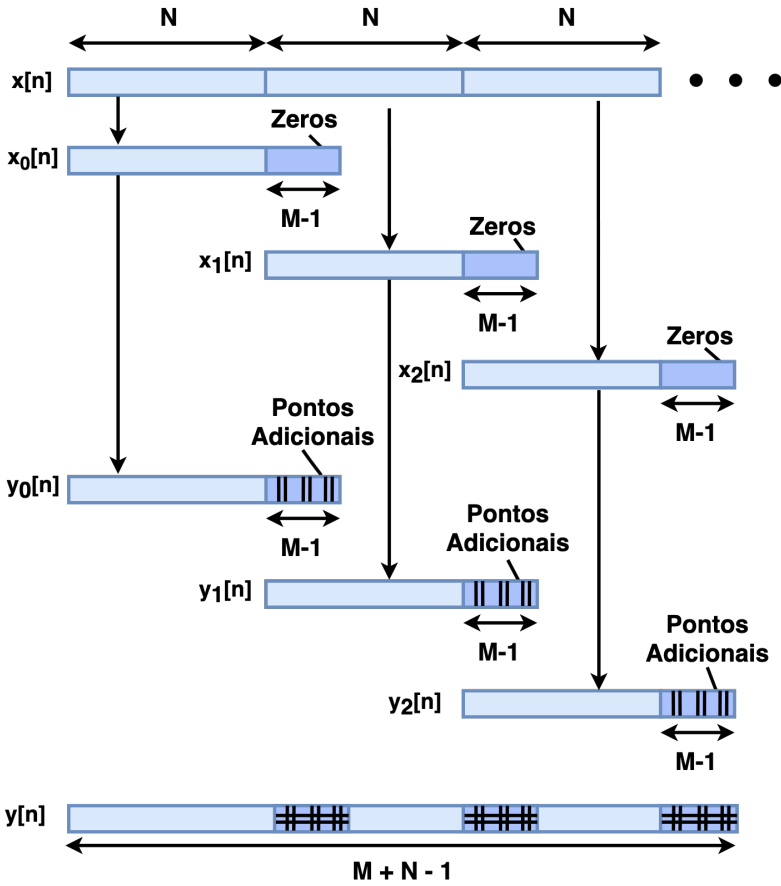
$$x_m[n] = \begin{cases} x[n + mN] & 0 \leq n \leq N - 1 \\ 0 & \text{demais } n. \end{cases} \quad (\text{A.6})$$

Substituindo (A.5) em (A.4):

$$\begin{aligned}
 y[n] &= \sum_{l=0}^{M-1} w[l] \sum_{m=0}^{\infty} x_m[n-l-mN] \\
 y[n] &= \left(\sum_{l=0}^{M-1} w[l] x_m[n-l-mN] \right) \\
 y[n] &= \sum_{m=0}^{\infty} y_m[n-mN].
 \end{aligned} \tag{A.7}$$

Cada subsequência $y_m[n]$ é a convolução de $x_m[n]$ e $w[n]$ com comprimento $M + N - 1$. Este cálculo também é possível usando DFTs ao estender as sequências $x_m[n]$ e $w[n]$ com $M - 1$ e $N - 1$ zeros, resultando em um $y_m[n]$ com comprimento de $M + N - 1$ amostras. Ao somar as subsequências $y_m[n]$ haverá uma sobreposição de $M - 1$ amostras, sendo este efeito que gera o resultado desejado e que nomeia o método como *overlap-add*. A Figura A.4 apresenta todo o processo descrito de maneira gráfica.

Figura A.4: Representação gráfica do método *overlap-add*



Fonte: do Autor.

A.3 Escala mel

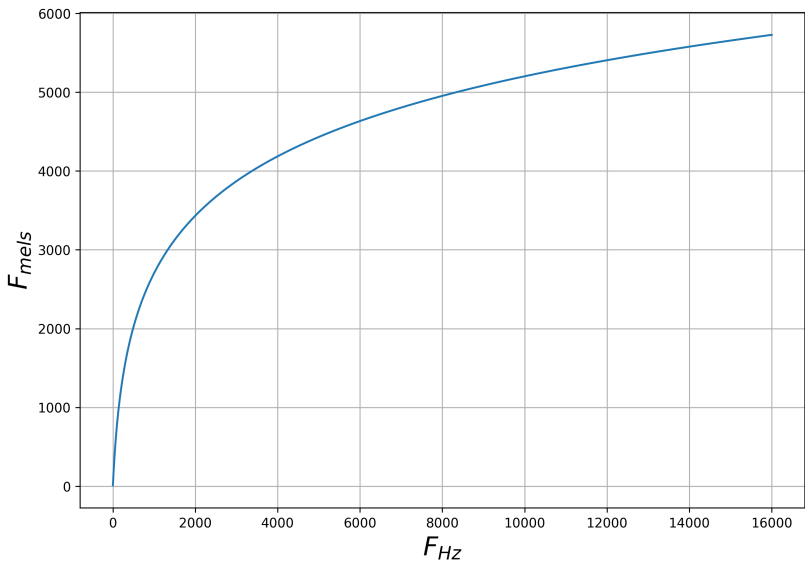
A escala *mel* [58] é uma escala perceptual não linear de frequências julgadas por ouvintes como equidistantes. Por se tratar de uma aproximação de um fenômeno psicofísico existem várias equações que tentam representar a escala. A equação mais utilizada em aplicações é derivada

de O'Shaughnessy [60]

$$m = 1127 \ln \left(1 + \frac{f}{100} \right) = 2595 \log_{10} \left(1 + \frac{f}{100} \right), \quad (\text{A.8})$$

onde m é a unidade de medida de frequências ou picos percebidos de um tom. Essa unidade não corresponde linearmente à frequência física, da mesma forma que o ouvido humano também não o faz. A Figura A.5 mostra as frequências em Hz representadas na escala *mel*, percebe-se que de 0 a 1kHz a resposta é aproximadamente linear, acima disso a escala se torna logarítmica.

Figura A.5: Relação entre escala linear e escala *mel* de frequência.



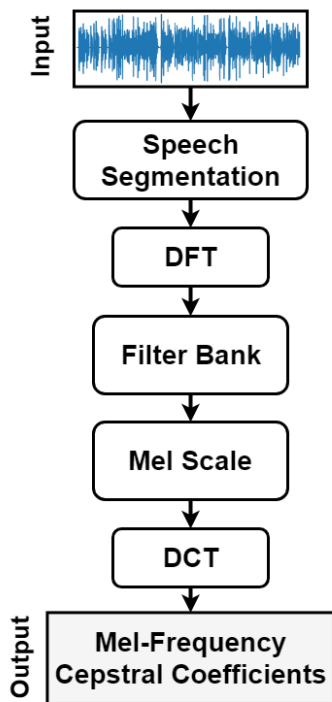
Fonte: do Autor.

Em sistemas que envolvem processamento de fala é corriqueira a utilização de coeficientes mel-cepstrais (*mel frequency cepstral coefficients* - MFCC) como forma de parâmetro de análise [10]. Esses coeficientes são uma representação definida como *cepstrum*, que apresentam uma boa representação das propriedades locais do espectro de um dado *frame* do sinal de fala [59].

Calcula-se os coeficientes MFCC ao aplicar um banco de filtros di-

gitalis sobre a magnitude de um espectrograma, logo após aplica-se o logaritmo de *mel* (A.8) na saída dos filtros, mapeando as magnitudes correspondentes a cada uma das frequências na escala *mel*. Toma-se a transformada discreta de cosseno (DCT), as amplitudes de saída correspondem aos MFCCs. A Figura A.6 corresponde ao diagrama de blocos do processo de cálculos dos coeficientes MFCC, como fora descrito.

Figura A.6: Diagrama de Blocos do cálculo do *Mel-frequency Cepstrum Coefficients*.

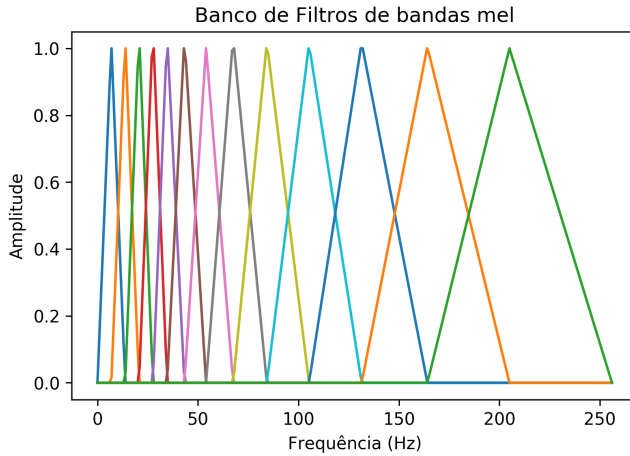


Fonte: do Autor.

A Figura A.7 apresenta o espaçamento não linear dos filtros para o cálculo dos coeficientes em 13 bandas *mel*. O banco de filtros é formado por filtros triangulares, espaçados de acordo com a escala de frequências de *mel*, ou seja, o espaçamento entre os filtros não é linear. Cada filtro calcula a média do espectro em torno da frequência central e tem

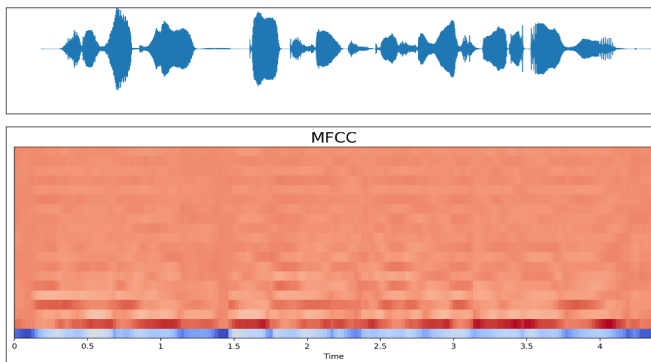
diferentes larguras de banda, sendo a largura de banda maior quanto mais alta for a frequência. A Figura A.8 apresenta a representação gráfica dos coeficientes MFCCs de um sinal de fala no decorrer do tempo.

Figura A.7: Resposta em frequências do banco de filtros de 13 bandas *mel*.



Fonte: do Autor.

Figura A.8: Parâmetros MFCC de um sinal de fala.



Fonte: do Autor.

Conceitos Básicos de Aprendizagem de Máquina

Aprendizagem de máquina é definida por Tom M. Mitchell [34] como: “Diz-se que um programa de computador aprende pela experiência E , com respeito a algum tipo de tarefa T e performance P , se sua performance P nas tarefas em T , na forma medida por P , melhoram com a experiência E ”.

B.1 Categorias de Aprendizagem de Máquina

Existem três categorias para as tarefas do aprendizado de máquina, cada uma de acordo com o tipo de sinal ou *feedback* que serão explorados. São elas [20]:

- **Aprendizado supervisionado:** Os dados são rotulados por um “professor”. Dessa forma a máquina aprende uma regra geral de mapeamento das entradas para as saídas.
- **Aprendizado não supervisionado:** Os dados de entrada não são rotulados. A máquina aprende a identificar padrões por si só, portanto é utilizado como um meio de descobrir novos padrões nos dados.

- **Aprendizagem por reforço:** A máquina interage com o meio de forma dinâmica, recebe um *feedback* com premiações ou punições à medida que desbrava o meio.

B.2 Sub-Categorias de Aprendizagem Supervisionada

A aprendizagem supervisionada é categorizada em dois problemas, classificação e regressão. Neste trabalho faremos uso apenas da classificação.

B.2.1 Classificação

A classificação é o processo de atribuir a alguma entrada um rótulo identificador, ou seja, tenta-se mapear variáveis de entrada em categorias diferentes.

A Figura B.1 mostra um exemplo de classificação binária, no qual a entrada \mathbf{x} possui duas dimensões (x_1, x_2). Por se tratar de uma classificação binária há apenas duas classes distintas, sendo elas representadas por triângulos e bolinhas. A reta pontilhada representa o limiar de decisão do classificador, a partir do qual determina-se a que classe cada elemento de \mathbf{x} pertence.

B.3 Sub-Categorias de Aprendizagem Não Supervisionada

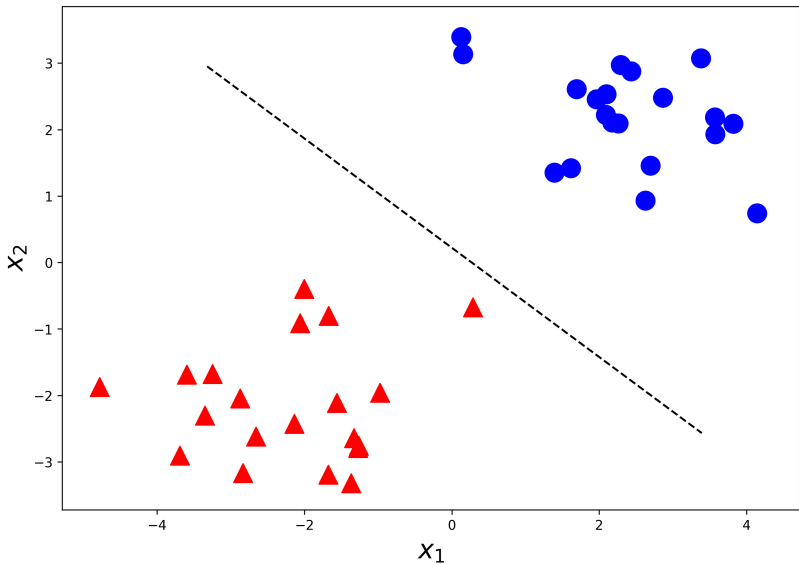
Uma das sub-categorias de aprendizagem não supervisionada é o *clustering*, método que segmenta os dados com padrões semelhantes em grupos (*clusters*). Existem diversas técnicas para realizar essa tarefa, todas elas procuram por semelhanças e diferenças num conjunto de dados, agrupando os semelhantes em *clusters*, de acordo com alguma métrica ou critério de decisão.

Neste trabalho utilizou-se duas técnicas de *clustering* o *K-means* e o *Spectral Clustering*.

B.3.1 *K-means*

Considere um conjunto de amostras (x_1, x_2, \dots, x_n) em um espaço multi-dimensional. Além disso, cada uma das N amostras corresponde a um

Figura B.1: Classificação Binária.

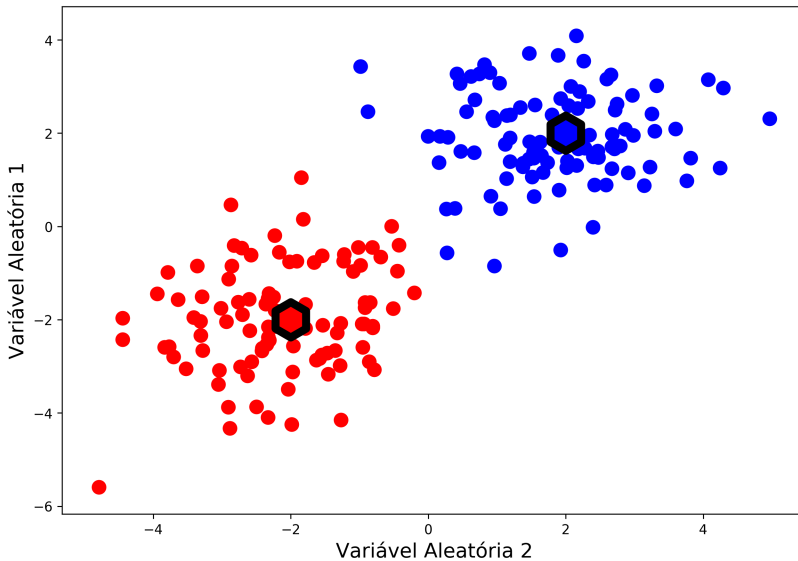


Fonte: do Autor.

vetor D -dimensional [6]. O algoritmo K -means particiona os N vetores em um número K de *clusters*, sendo K um valor pré-definido. Intuitivamente associa-se um *cluster* como um grupo de dados que possui distância entre as amostras do mesmo grupo inferior quando comparadas com a distância de amostras de outro grupo.

Define-se um conjunto de vetores D -dimensionais μ_k , com $k = 1, \dots, K$, onde cada μ_k corresponde a um centróide associado ao k^{th} *cluster*. O objetivo do algoritmo é encontrar um agrupamento para as amostras, categorizadas por um conjunto de centróides, de modo que a soma dos quadrados das distâncias entre as amostras e o centróide mais próximo seja mínima.

A Figura B.2 mostra graficamente o agrupamento com dois *clusters* ($K = 2$). Observa-se um padrão de agrupamento entre as amostras em torno de hexágonos, representando os centróides. Formalmente define-se, para cada amostra de dados x_n , um indicador binário $r_{nk} \in \{0, 1\}$, que caso x_n é agrupado a um *cluster* k , assim $r_{nk} = 1$, e $r_{nj} = 0$ para $j \neq k$.

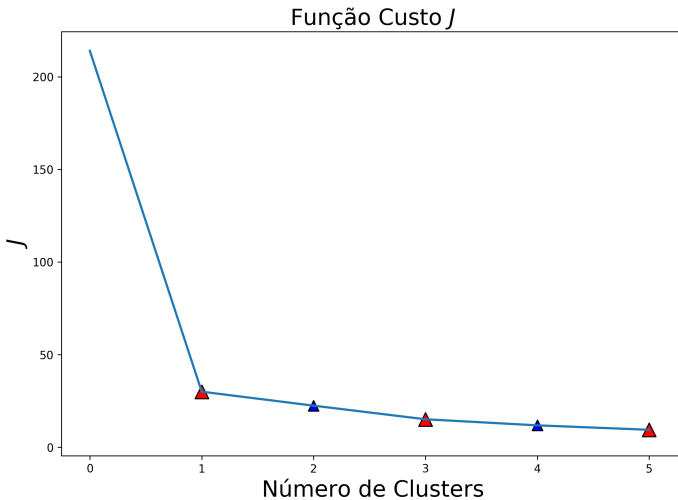
Figura B.2: *K-means Clustering* com $K = 2$.

Fonte: do Autor.

O procedimento de otimização da função custo, definida como

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|^2, \quad (\text{B.1})$$

é dado em dois estágios [6]. No primeiro os valores de μ_k são mantidos fixos e minimiza-se J em função de r_{nk} . No segundo fixa-se os valores de r_{nk} e minimiza-se J em função de μ_k . Repete-se os dois estágios até que J atinja a convergência. A Figura B.3 apresenta um caso de otimização com 5 *clusters* para melhor visualização da convergência da função ocorrendo. Os pontos triangulares são denominados de *elbow point*, usualmente representados onde se inicia os retornos decrescentes ao aumentar o número de *clusters* K .

Figura B.3: Representação gráfica da convergência da função custo J .

Fonte: do Autor.

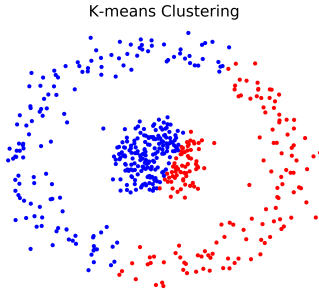
B.3.2 Spectral Clustering

A clusterização utilizando K -means apresenta limitações quando a geometria dos *clusters* é mais complicada, devido o algoritmo sempre decidir os agrupamentos por uma função linear. Isto leva a uma falha quando se necessita de agrupamentos de dados com fronteiras mais complicadas. Uma alternativa para contornar esta limitação é utilizar métodos de clusterização espectral.

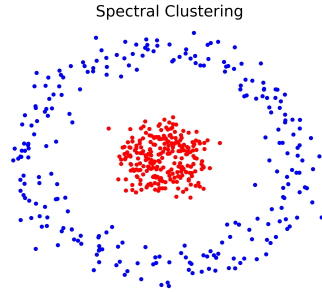
O *Spectral Clustering* é uma técnica que faz uso dos autovetores de matrizes derivadas da distância entre pontos de um conjunto de dados [37]. Essa técnica reduz a dimensionalidade dos dados antes de se executar a clusterização. As amostras de entrada são tratadas como os nodos de um grafo, implicando em um problema de partição de grafos. Estes nodos são mapeados em um espaço dimensional reduzido. Uma comparação entre os K -means e o *Spectral Clustering* sobre um mesmo conjunto de dados é apresentada na Figura B.4, devido a geometria do conjunto de dados percebe-se a deficiência do K -means em efetuar a clusterização, deficiência essa devido ao fato da técnica utilizar um decisor linear.

Figura B.4: Comparação entre *K-means* e *Spectral Clustering*.

- (a) *K-means* de um conjunto de dados circulares. (b) *Spectral Clustering* de um conjunto de dados circulares.



Fonte: do Autor.



Fonte: do Autor.

O processo do *spectral clustering* utilizado neste trabalho é uma variação do método clássico [37], proposta em [66] e que apresenta um refinamento na matriz de similaridade gerada. O algoritmo proposto em [66] consiste nos seguintes passos:

Dado um conjunto de pontos $S = \{s_1, \dots, s_n\} \in \mathbb{R}^l$ é construída a matriz de similaridade $A \in \mathbb{R}^{n \times n}$, definida por

$$A_{ij} = \exp(-\|s_i - s_j\|^2 / \sigma^2), \quad \text{para } i \neq j \text{ e } A_{ii} = 0, \quad (\text{B.2})$$

e os valores máximos de cada linha da matriz são colocados na diagonal principal ($A_{ii} = \max_{j \neq i} A_{ij}$). Com a matriz de similaridade pronta aplica-se uma sequência de operações de refinamento sobre a mesma, ao tratá-la como uma imagem. Sendo as operações:

- (i) *Gaussian Blur*, reduz os efeitos de *outliers* suavizando as bordas dos dados;
- (ii) *Row-wise Thresholding*, zera as afinidades entre as *features* de classes distintas;
- (iii) Simetrização, restaura a simetria da matriz de similaridade ($Y_{ij} = \max(X_{ij}, X_{ji})$), crucial para efetuar a clusterização;
- (iv) Difusão, refina a imagem de tal forma que as fronteiras das seções que diferenciam as classes fiquem mais evidentes ($Y = XX^T$);

- (v) *Row-wise Max Normalization*, reescala o espectro para garantir que não ocorram efeitos de escalas indesejadas durante a clusteração ($Y_{ij} = X_{ij}/\max_k X_{ik}$).

Após o refinamento da matriz de similaridade usa-se a decomposição de valores singulares [19]. Sendo os n autovalores $\lambda_1 > \lambda_2 > \dots > \lambda_n$, o número de *clusters* \tilde{k} é definido como:

$$\tilde{k} = \arg \max_{1 \leq k \leq n} \frac{\lambda_k}{\lambda_{k+1}}. \quad (\text{B.3})$$

B.4 Deep Learning

Algoritmos baseados em *deep learning* vem ganhando força em cenários como reconhecimento de fala [10, 29] e processamento de linguagem natural [54]. Isso deve-se ao fato de que redes profundas são capazes de extrair cada vez mais características de áudio devido a sua grande quantidade de camadas e unidades ocultas, ou seja, é possível modelar uma função altamente não linear a partir do conjuntos de funções não lineares mais simples [44, 46].

B.4.1 Multilayer Perceptrons (MLPs)

A MLP é uma rede neural de alimentação direta (*feedforward network*), composta por camadas de neurônios interligados por sinapses com pesos [20]. A terminologia *feedforward* deve-se às informações fluírem diretamente pela função de aproximação, portanto não há realimentação. Quando uma rede *feedforward* possui realimentação usa-se o termo *Recurrent Neural Networks* (RNNs), apresentadas na Seção B.4.3.

O objetivo de uma rede MLP é aproximar uma função f^* qualquer, que será modelada por uma composição de funções $y = f(x, W)$ cujos parâmetros são aprendidos no processo de treinamento da rede. Cada função é representada por uma camada, e a interconexão das mesmas forma uma rede.

Em cada camada os sinais de entrada são multiplicados por um peso correspondente W_n . A combinação linear dos valores de entrada ponderados somados ao um vetor de *bias* passam por uma função de ativação. A saída da função de ativação corresponde à saída da camada

expressa por:

$$y = f(b + x_1 W_1 + \dots + x_n W_n) = f\left(b_0 + \sum_{i=1}^N x_i * W_i\right). \quad (\text{B.4})$$

Seja $y = f^*(x)$ uma função que se deseja aproximar, a modelagem da rede com N camadas que prevê um \hat{y} é definida por:

$$f(x) = f_N(f_{N-1}(\dots f_2(f_1(x)))) = \hat{y}. \quad (\text{B.5})$$

A Figura B.5 corresponde a uma representação arquitetural de uma rede neural MLP. Cada círculo corresponde a uma camada, o círculo amarelo representa o vetor de *bias*, as arestas que interconectam as camadas representam os pesos distribuídos.

B.4.2 Funções de Ativação

A função de ativação é um operador não linear que mapeia os pesos de entrada para as saídas de cada neurônio na rede. Ou seja, definem se a informação que o neurônio está recebendo é ou não relevante. A escolha da função depende do tipo de dados e de tarefa que a rede deve executar.

Por exemplo, a logística sigmoide, ou logística, é uma função suave e continuamente diferenciável sendo ela e sua derivada representadas por:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad \sigma'(x) = \sigma(x)(1 - \sigma(x)). \quad (\text{B.6})$$

A função logística é apenas sensível para valores de x próximos de zero, para valores mais elevados a saída da sigmoide satura em ± 1 . Devido a essa característica podem ocorrer dificuldades na otimização durante o treinamento em redes *feedforward* [20, 21].

Por sua vez, a ativação ReLU (*Rectified Linear Unit*) é extremamente utilizada por sua facilidade de otimização e por não apresentar problemas de saturação como a logística. A ReLU é definida como:

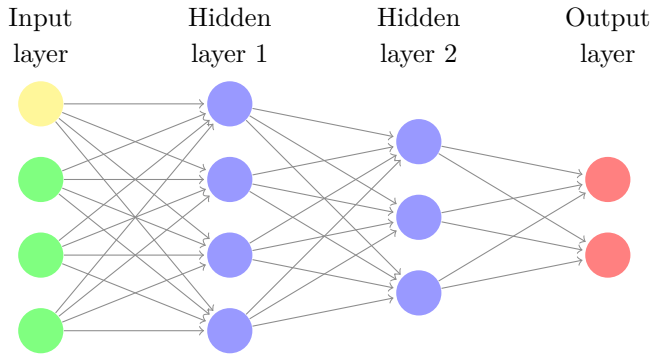
$$\text{ReLU}(x) = \max(0, x). \quad (\text{B.7})$$

Para problemas de classificação uma função útil para lidar com múltiplas classes é a *softmax*. A função *softmax* é um tipo de função sig-

moide, porém diferentemente da logística consegue lidar com mais do que apenas duas classes [6]. É definida pela generalização da função logística para $K > 2$ como:

$$\text{softmax}(x_k) = \frac{e^{x_k}}{\sum_{k=1}^K e^{x_k}}. \quad (\text{B.8})$$

Figura B.5: Exemplo de uma rede neural MLP com duas camadas ocultas.



Fonte: do Autor.

B.4.3 Redes Neurais Recorrentes (RNNs)

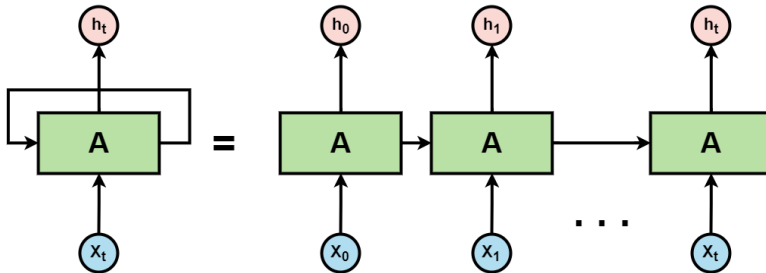
A camada oculta de uma rede neural *feedforward* recebe sinais apenas da camada de entrada, os processa e passa para a camada de saída. Uma rede recorrente (RNN) difere nesse ponto, pois a camada oculta recebe os sinais provenientes da camada de entrada e da camada de saída da iteração anterior, ou seja, há uma realimentação.

Pela sua realimentação as RNNs funcionam como uma memória, podendo armazenar informações ao processar novas entradas [20]. A cada período a rede não só armazena informações desse instante como também de instantes anteriores. Essa memória torna as RNNs ideais para tarefas que envolvam o processamento de dados representados por séries temporais.

A Figura B.6 mostra um diagrama de blocos de uma rede neural recorrente, define-se A como um pedaço da rede, x_t um sinal de entrada e h_t os valores de saída da rede. A representação da funcionalidade da rede no tempo é apresentada ao abrir o laço em alguns instantes, dessa

forma a saída computada no instante de $t = 0$ influenciará a saída no próximo instante assim que a rede executar uma nova entrada, e assim sucessivamente para os demais instantes.

Figura B.6: Rede Neural Recorrente com laço desenrolado.



Fonte: Christopher Olah [39].

As RNNs possuem quatro formas distintas de processar as sequências de entrada; a forma de leitura e a geração de sequências de saída dependem da arquitetura da rede. Essas arquiteturas são divididas em quatro classes, sendo elas: *i)* *One to one*, que não difere em nada de uma rede *feedforward* clássica; *ii)* *One to many*, o dado de entrada é lido apenas uma vez para o estado oculto, a informação gerada é passada adiante entre os estados em diferentes períodos de tempo, além de cada estado oculto gerar uma saída; *iii)* *Many to one*, nessa arquitetura os dados de entrada são lidos a cada instante de tempo, porém uma única saída é gerada ao finalizar a leitura de todos dados de uma sequência; *iv)* *Many to many*, há duas formas de implementação dessa arquitetura. Na primeira os dados da sequência são lidos em períodos de tempo e não são calculadas as suas saídas, ou seja, provocasse uma defasagem entra a leitura e a saída da sequência. Na segunda forma a cada período de tempo, dada uma entrada de dados de uma sequência, é gerada uma saída correspondente.

A Figura B.7 mostra o diagrama para cada forma arquitetural de uma rede RNN. Os círculos azuis e vermelhos são respectivamente os vetores de entrada e as saídas da rede, os retângulos verde representam os vetores que armazenam os valores gerados pelos estados ocultos.

A formulação matemática de uma RNN, em função do *bias* b , pesos

W e entradas x pode ser descrita como

$$\begin{aligned} h_t &= f(b_h + X_t W_x + h_{t-1} W_h) \\ \hat{y}_t &= b_0 + h_t W_0, \end{aligned} \tag{B.9}$$

A diferença de (B.9) em comparação a uma rede *feedforward* (B.4), deve-se à adição da informação do estado oculto do período anterior ($h_{t-1} W_h$). Essa informação se propaga pela rede, ao abrir o laço percebe-se a conexão entre os estados ocultos, como mostra a Figura B.6. O equacionamento, por exemplo, de 3 períodos de tempo de uma RNN *many to one* é descrito como

$$\begin{aligned} h_0 &= f(b_h + X W_x) \\ h_1 &= f(b_h + X W_x + h_0 W_h) \\ h_2 &= f(b_h + X W_x + h_1 W_h) \\ \hat{y}_t &= b_0 + h_3 W_0 \end{aligned} \tag{B.10}$$

Devido a essa característica de memória, as RNNs podem apresentar problemas no treinamento, por conta das dependências de longas seqüências (*long-term dependencies*) [5]. Em muitos casos práticos a rede não consegue lidar com essas dependências, devido ao erro desaparecer em métodos de treinamento baseado em gradientes [41].

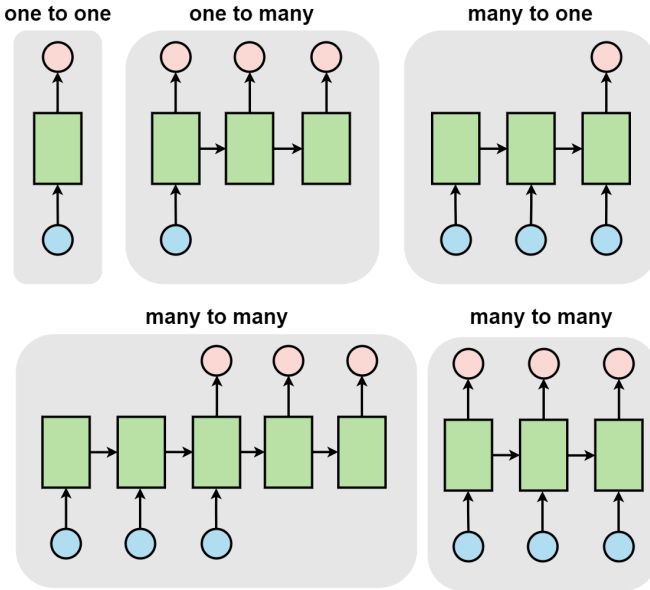
B.4.3.1 Long Short-Term Memory (LSTM)

Propostas por Hochreiter & Schmidhuber em 1997 [24], as redes *Long Short-Term Memory* (LSTM) foram introduzidas para resolver o problema de desaparecimento de gradiente que RNNs introduzem devido a grandes seqüências de dados.

A correção deste problema é feita ao manter o fluxo do erro constante usando unidades especiais chamadas de *gates*. Os *gates* são compostos por uma camada sigmoide e um multiplicador ponto-a-ponto. A camada sigmoide gera como saída um número no intervalo [0, 1]. Este valor corresponde à porcentagem que deve-se descartar de cada componente, por exemplo, caso seja zero a componente é totalmente descartada.

A Figura B.8 mostra a estrutura da rede LSTM, contendo os fluxos dos sinais, onde x_t é a seqüência de entrada e h_t corresponde ao vetor

Figura B.7: Tipos de Arquiteturas de Redes Neurais Recorrentes.



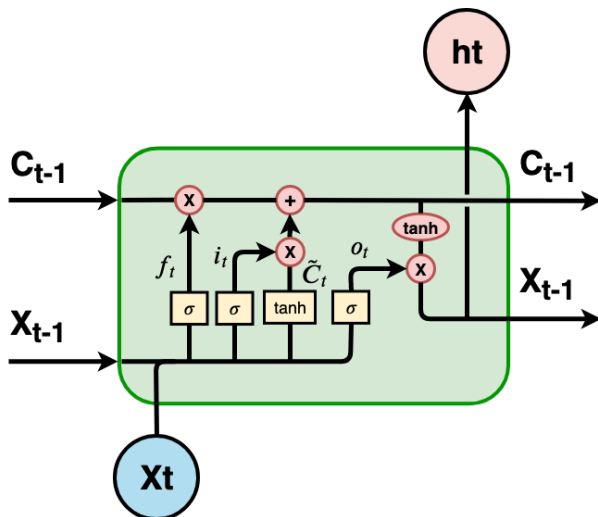
Fonte: Christopher Olah [39].

do estado oculto. Os vetores de ativação i_t , f_t e o_t são respectivamente relacionados aos *gates* de entrada, esquecido e de saída. W corresponde aos pesos calculados em cada estado oculto e C_t corresponde à célula de estado.

Assim, uma rede LSTM pode ser formalmente definida como [24]

$$\begin{aligned}
 C_t &= f_t \times C_{t-1} + i_t \times \tilde{C}_t \\
 \tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\
 h_t &= o_t \times \tanh(C_t) \\
 o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
 i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
 f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)
 \end{aligned} \tag{B.11}$$

Figura B.8: Topologia da rede LSTM.



Fonte: Christopher Olah [39].

B.4.4 Extração de Parâmetros

Em tarefas de processamento de fala são comumente utilizados dois super-vetores diferentes como forma de parâmetros de classificação, conhecidos como *i-vectors* (*identify vectors*) [14] e *d-vectors* (*deep vectors*) [62]. Ambos vetores representam um identificador dentro de um espaço de locutores, dessa forma são conhecidos como “vetores de locutor” (*speaker vectors*) [29].

Fundamentalmente os dois vetores são diferentes, apesar de utilizados para a mesma tarefa. Os *i-vectors* são baseados no modelo Gaussiano linear, para o qual aprendizado é não supervisionado com critério de aprendizado sendo a máxima verossimilhança das *features* acústicas. Em contraste, os *d-vectors* são baseados em redes neurais profundas (DNNs), dessa forma o aprendizado é supervisionado com critério de aprendizado sendo a máxima distinção entre locutores [29, 62].

Ambos parâmetros estão presentes nas aplicações do estado da arte para *Speaker Diarization*, seja utilizando *Gaussian Mixture Modelling* (GMM) para extração de *i-vectors*, ou a extração dos *d-vectors* utilizando redes neurais, com o último sendo o foco deste trabalho.

Os *d-vectors* são derivados dos parâmetros de saída da ativação da última camada oculta, ou seja, antes da camada de saída. Para o cálculo do parâmetro *d-vector* \vec{d} a uma dada entrada x^t , calcula-se as saídas das ativações h_j^t da última camada oculta (j). A partir dos valores de saída das ativações pode-se calcular \vec{d} como [9]

$$\vec{d} = \max_t (h_j^t). \quad (\text{B.12})$$

Bases de Dados Utilizadas

Diferentes bases de dados foram utilizadas para o desenvolvimento do sistema implementado. O modelo da rede LSTM [30] utilizado foi treinado com três bases de dados distintas: VCTK [11], LibriSpeech [40] e VoxCeleb [12, 36]. Para efetuar a validação do sistema completo utilizou-se a base de dados MUSAN, usada para avaliação de generalidade. Todas as bases de dados citadas são gratuitas, tem fácil acesso e usabilidade.

C.1 VCTK Corpus

A base VCTK, montada pelo CSTR (*Centre for Speech Technology Research*) [11], contém dados de 109 locutores nativos da língua inglesa, variando os sotaques dependendo do país ou região. Cada locutor possui 400 sentenças gravadas, das quais grande parte eram notícias de jornais lidas por cada um dos locutores. As gravações foram todas efetuadas de mesma forma para todas as amostras coletadas, utilizando-se um microfone unidirecional em uma câmara semi anecoica na Universidade de Edinburgh. Essas gravações foram disponibilizadas com duas configurações diferentes: a primeira sendo a versão RAW (sem com-

pressão) com 96kHz de frequência de amostragem com quantização de 24 bits; a segunda versão, com compressão, possui frequência de amostragem de 48kHz e quantização de 16 bits.

C.2 LibriSpeech ASR Corpus

O LibriSpeech, é uma base de dados montada para aplicações envolvendo reconhecimento automático de fala (*ASR - Automatic Speech Recognition*), possuindo mais de 1000 horas de fala proveniente de *audiobooks* gratuitos, todos amostrados a 16kHz. O LibriSpeech é dividido em pequenos outros conjuntos (*subsets*), apresentados na Tabela C.1. Cada *subset* é usado para um diferente propósito, sendo dividido em três categorias: *train*, *test* e *dev*. Os *subsets* iniciando com *train* são os utilizados para treinar os modelos (HMMs, GMMs ou Redes Neurais), os que iniciam com *test* contém amostras dos mesmo locutores do *subset* *train* para verificar o treinamento do modelo, e por fim os *subsets* que começam com a marcação *dev* são locutores desconhecidos pelo modelo, utilizados para avaliação de generalidade da rede.

Tabela C.1: Divisão em *subsets* da base LibriSpeech

Subset	Horas	Minutos por Locutor	Locutores Femininos	Locutores Masculinos	Total de Locutores
dev-cle1n	5.4	8	20	20	40
test-clean	5.4	8	20	20	40
dev-other	4.3	10	16	17	33
test-other	5.1	10	17	16	33
train-clean-100	100.6	25	125	126	251
train-clean-360	363.6	25	439	482	921
train-other-500	496.7	30	564	602	1 166

Fonte: *LibriSpeech: An ASR Corpus based on Public Domain Audio Books* [40].

C.3 VoxCeleb

A base de dados VoxCeleb possui duas versões [12, 36], ambas bases utilizam sinais de fala de celebridades, extraídos de vídeos disponíveis no YouTube. A primeira versão, chamada de VoxCeleb1 possui cerca

de 100.000 sentenças de fala de 1 251 celebridades, sendo cerca de 55% locutores masculinos. A segunda versão, o VoxCeleb2, possui mais de 1 milhão de sentenças de mais de 6 mil celebridades, com 61% dos locutores do sexo masculino.

Ambas versões apresentam variedade de etnias, sotaques, profissões e idade, com essas características de cada celebridade extraídas do Wikipedia. A Tabela C.2 mostra alguns dados relevantes da base, como o número total de locutores, quantas horas de fala e a duração média de cada sentença.

Tabela C.2: Estatísticas das bases VoxCeleb1 e VoxCeleb2.

	VoxCeleb1	VoxCeleb2
Locutores	1 251	6 112
Locutores Masculinos	690	3 761
Tamanho da Base em Horas	352	2 442
Sentenças de Fala	153 516	1 128 246
Duração Média em Segundos das Sentenças	8.2	7.8

Fonte: *VoxCeleb2: Deep Speaker Recognition* [12].

C.4 MUSAN: A Music, Speech, and Noise Corpus

MUSAN é uma base montada principalmente para atender duas tarefas, treinar modelos para *Voice Activity Detection* (VAD) e discriminação entre música e fala [57]. A base possui aproximadamente 109 horas de áudio de domínio público, possuindo três categorias: *music* (música), *speech* (fala) e *noise* (ruído). Todos os arquivos têm frequência de amostragem de 16kHz, e possuem anotações, por exemplo, para música são marcadas os intervalos de tempo com a presença ou ausência de vocais e o gênero da pessoa. Os arquivos de interesse para este trabalho são os da categoria *speech*. Essa partição possui cerca de 60 horas de arquivos de fala, sendo 20 horas provenientes de *audiobooks* e o restante são arquivos do governo dos EUA de debates, comitês e discursos. Todos

arquivos da base são em língua inglesa.

Alguns dados importantes para métricas de avaliação são disponibilizados em [57]. Ao utilizar-se um VAD baseado apenas em energias de *frames*, um aumento na taxa de erro de sistemas *Speaker Recognition* é perceptível. Um estudo comparativo entre utilizar VAD baseado apenas em energia *versus* GMM com energia, é apresentado em [57]. Neste estudo são calculadas as taxas de erro EER para diferentes tamanhos de janelas, percebe-se que quanto menor for a janela de tempo do sinal de fala, maior é a EER. Além disto, um comparativo de melhoria de desempenho é calculada, demonstrando que o uso de um VAD mais robusto pode-se ter uma melhoria de até 20 vezes.