UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

SÉRGIO MONTAZZOLLI SILVA

Redução de Dimensionalidade Aplicada à Diarização de Locutor

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Dr. Dante Augusto Couto Barone
Orientadora

Prof. Dr. André Gustavo Adami
Co-orientador

Porto Alegre, novembro de 2013

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Silva, Sérgio Montazzolli

Redução de Dimensionalidade Aplicada à Diarização de Locutor / Sérgio Montazzolli Silva. – Porto Alegre: PPGC da UFRGS, 2013.

81 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2013. Orientadora: Dante Augusto Couto Barone; Co-orientador: André Gustavo Adami.

1. Diarização de Locutor. 2. Análise de Discriminantes. 3. Redução de Dimensionalidade. I. Barone, Dante Augusto Couto. II. Adami, André Gustavo. III. Título.

“The temptation to form premature theories upon insufficient data is the bane of our profession.”

— SHERLOCK HOLMES, PERSONAGEM FICTÍCIO.

AGRADECIMENTOS

Primeiramente agradeço as pessoas que colaboraram diretamente, ao longo de todos os anos da minha vida, para este dia chegar: meus pais Gerson e Sueli, e minha irmã Júlia.

Agradeço aos meus orientadores, prof. Dante Barone e prof. André Adami, por toda a paciência e tempo dedicado ao longo destes mais de 2 anos de curso, para me corrigir e ensinar o caminho certo. Sem a ajuda de ambos, a realização deste trabalho não seria possível. Em especial, quero agradecer ao prof. Adami por despender horas e horas de seu tempo, mesmo em feriados, finais de semana e horários noturnos, para reuniões via Skype, que foram importantíssimas para lapidar meu conhecimento.

Ha minha família, minhas avós Lady e Maria, e meu avôs Helio e Olímpio, a todos os meus tios, tias, e primos. Aos meus amigos de Londrina, Ibaiti e Porto Alegre, por terem me aguentado reclamando nos momentos difíceis, mas principalmente por terem me proporcionado os momentos mais alegres.

Também sou grato a Universidade Estadual de Londrina e ao corpo docente do Departamento de Computação, que construíram a base do meu conhecimento.

Finalmente, agradeço ao Instituto de Informática (INF) e a Universidade Federal do Rio Grande do Sul, por todo o suporte, infraestrutura e matérias de altíssima qualidade oferecidas.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

AHC	*Agglomerative Hierarchical Clustering*
AMI	*Augmented Multi-party Interaction*
BAT	Distância de Bhattacharyya
BIC	*Bayesian Information Criterion*
DER	*Diarization Error Rate*
DL	Diarização de Locutor
DTFT	*Discrete-Time Fourier Transform*
E-HMM	*Evolutive Hidden Markov Models*
EM	*Expectation–Maximization*
FlsD	*Fisher Linear Semi-Discriminant Analysis*
FST	*False-alarm Speaker Time*
GLR	*Generalized Likelihood Ratio*
GMM	*Gaussian Mixture Moedels*
GSM	*Global System for Mobile Communications*
HMM	*Hidden Markov Models*
ICR	*Information Change Rate*
JT	Janelas de Textura
KL2	*Kullback-Leibler Divergence*
LDA	*Linear Discriminant Analysis*
LPC	*Linear Predictive Coding*
LPCC	*Linear Predictive Cepstrum Coeficients*
LSP	*Line Spectral Pairs*
MEL	*Mel Spectrum Coefficients*
MFCC	*Mel-Frequency Cepstrum Coefficients*
ML	*Machine Learning*
MST	*Missed Speaker Time*

NIST	*National Institute of Standards and Technology*
PCA	*Principal Component Analysis*
PLP	*Perceptually Linear Predictive*
RT	*Rich Transcription*
SAD	*Speech Activity Detection*
SDL	Sistema de Diarização de Locutor
SET	*Speaker Error Time*
SRE	*Speaker Recognition Evaluation*
SVM	*Support Vector Machine*
TE	Tempo de Execução
UBM	*Universal Background Model*
VAD	*Voice Activity Detection*

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Atualmente existe uma grande quantidade de dados multimídia sendo geradas todos os dias. Estes dados são oriundos de diversas fontes, como transmissões de rádio ou televisão, gravações de palestras, encontros, conversas telefônicas, vídeos e fotos capturados por celular, entre outros. Com isto, nos últimos anos o interesse pela transcrição de dados multimídia tem crescido, onde, no processamento de voz, podemos destacar as áreas de Reconhecimento de Locutor, Reconhecimento de Fala, Diarização de Locutor e Rastreamento de Locutores. O desenvolvimento destas áreas vem sendo impulsionado e direcionado pelo NIST, que periodicamente realiza avaliações sobre o estado-da-arte. Desde 2000, a tarefa de Diarização de Locutor tem se destacado como uma das principáis frentes de pesquisa em transcrição de dados de voz, tendo sido avaliada pelo NIST por diversas vezes na última década. O objetivo desta tarefa é encontrar o número de locutores presentes em um áudio, e rotular seus respectivos trechos de fala, sem que nenhuma informação tenha sido previamente fornecida. Em outras palavras, costuma-se dizer que o objetivo é responder a questão "Quem falou e quando?". Um dos grandes problemas nesta área é se conseguir obter um bom modelo para cada locutor presente no áudio, dada a pouca quantidade de informações e a alta dimensionalidade dos dados. Neste trabalho, além da criação de um Sistema de Diarização de Locutor, iremos tratar este problema mediante à redução de dimensionalidade através de análises estatísticas. Usaremos a Análise de Componentes Principáis, a Análise de Discriminantes Lineares e a recém apresentada Análise de Semi-Discriminantes Lineares. Esta última utiliza um método de inicialização estático, iremos propor o uso de um método dinâmico, através da detecção de pontos de troca de locutor. Também investigaremos o comportamento destas análises sob o uso simultâneo de múltiplas parametrizações de curto prazo do sinal acústico. Os resultados obtidos mostram que é possível preservar - ou até melhorar - o desempenho do sistema, mesmo reduzindo substâncialmente o número de dimensões. Isto torna mais rápida a execução de algoritmos de Aprendizagem de Máquina e reduz a quantidade de memória necessária para armezenar os dados.

Palavras-chave: Diarização de Locutor, Análise de Discriminantes, Redução de Dimensionalidade.

Dimensionality Reduction Applied to Speaker Diarization

ABSTRACT

Currently, there is a large amount of multimedia data being generated everyday. These data come from various sources, such as radio or television, recordings of lectures and meetings, telephone conversations, videos and photos captured by mobile phone, among others. Because of this, interest in automatic multimedia data transcription has grown in recent years, where, for voice processing, we can highlight the areas of Speaker Recognition, Speech Recognition, Speaker Diarization and Speaker Tracking. The development of such areas is being conducted by NIST, which periodically promotes state-of-the-art evaluations. Since 2000, the task of Speaker Diarization has emerged as one of the main research fields in voice data transcription, having been evaluated by NIST several times in the last decade. The objective of this task is to find the number of speakers in an audio recording, and properly label their speech segments without the use of any training information. In other words , it is said that the goal of Speaker Diarization is to answer the question "Who spoke when?". A major problem in this area is to obtain a good speaker model from the audio, given the limited amount of information available and the high dimensionality of the data. In the current work, we will describe how our Speaker Diarization System was built, and we will address the problem mentioned by lowering the dimensionality of the data through statistical analysis. We will use the Principal Component Analysis, the Linear Discriminant Analysis and the newly presented Fisher Linear Semi-Discriminant Analysis. The latter uses a static method for initialization, and here we propose the use of a dynamic method by the use of a speaker change points detection algorithm. We also investigate the behavior of these data analysis techniques under the simultaneous use of multiple short term features. Our results show that it is possible to maintain - and even improve - the system performance, by substantially reducing the number of dimensions. As a consequence, the execution of Machine Learning algorithms is accelerated while reducing the amount of memory required to store the data.

Keywords: Speaker Diarization, Discriminant Analysis, Dimensionality Reduction.

1 INTRODUÇÃO

Atualmente com o crescimento do volume de dados multimídia produzidos por fontes diversas (e.g. mídias online, chamadas telefônicas, transmissões de rádio e televisão, entre outros) surge a necessidade de métodos automáticos para extrair metadados que descrevam informações relevantes, como por exemplo: identificação dos locutores, significado semântico das sentenças, localização das fontes sonoras, entre outros. Esta tarefa é denominada, em Inglês, *Rich Transcription* (RT) (FURUI et al., 2012).

Como parte das tarefas que englobam a RT, podemos citar (ZWEIG; MAKHOUL; STOLCKE, 2006):

- *Reconhecimento de Fala*: traduz palavras faladas em texto;
- *Reconhecimento de Locutor*: encontra a identidade de um locutor a partir de segmentos acústicos;
- *Diarização de Locutor*: encontra locutores em um áudio de modo não-supervisionado, e indica seus respectivos trechos de fala;
- *Rastreamento de Locutores*: busca por trechos de fala pertencentes a um locutor conhecido;
- *Integração de Informações Multimodais*: uso de informações acústicas e visuais (vídeos) nas tarefas anteriores.

O presente trabalho trata da tarefa de Diarização de Locutor (DL). Esta tarefa é facilmente descrita pela resposta à pergunta "Quem falou e quando?" (TRANTER; REYNOLDS, 2006; ANGUERA MIRO et al., 2012). Atualmente é uma das mais importantes e desafiantes frentes de pesquisa em RT, dada a dificuldade de se reconhecer locutores sem nenhuma informação *a priori*, ou seja, tanto a identidade quanto o número de locutores presentes em um áudio são desconhecidos. A Figura 1.1 mostra a entrada e a saída de um Sistema de Diarização de Locutor (SDL). A entrada recebe apenas um sinal de onda discretizado que constitui um áudio, este sinal é processado pelo sistema, que interpreta as informações contidas nele. Na saída é gerado um conjunto de rótulos indicando os segmentos de fala de cada locutor encontrado, e também os locais de não-fala.

A partir do início dos anos 2000, a tarefa de DL, bem como as demais tarefas de RT, foram impulsionadas por avaliações de desempenho do Instituto Nacional de Padrões e Tecnologia dos Estados Unidos (*National Institute of Standards and Technology* - NIST). Desde então, várias bases de áudio foram criadas com o propósito de avaliar sistemas de RT. Os primeiros esforços de avaliar sistemas de DL pelo NIST datam de 2000 e 2002 durante as Avaliações de Reconhecimento de Locutor (*Speaker Recognition Evaluation* -

Figura 1.1: Entrada e saída de um Sistema de DL.

SRE) [1]. A partir de 2002 foi criada a Avaliação de RT[2], que engloba DL e várias outras tarefas de RT. Atualmente importantes conferências internacionais como a *International Conference ond Acoustics, Speech and Signal Processing* (ICASSP) e a *Conference of International Speech Communication Association* (INTERSPEECH), e periódicos como o *IEEE Transactions on Audio, Speech, and Language Processing*, possuem seções especiais para RT[3].

1.1 Objetivos e Motivação

Um dos grandes desafios em DL é criar um bom modelo que represente cada um dos locutores presentes em um áudio. Um áudio típico das bases do NIST, utilizadas neste trabalho, possui entre 2 e 5 minutos de duração, e pode conter de 2 a 9 locutores. Este tempo pode ser curto para caracterizar cada locutor presente. Soma-se a isso, a alta dimensionalidade dos dados após a aplicação de alguma técnica de parametrização de sinal, diga-se entre 12 e 24 dimensões. Muitas vezes é possível atenuar este problema com o uso de técnicas de redução de dimensionalidade, como a Análise de Componentes Principais ou a Análise de Discriminantes Lineares. Neste trabalho descreveremos como se comportam estas duas técnicas quando aplicadas ao problema em questão.

Em 2012, Giannakopoulos e Petridis propuseram um método chamado Análise de Semi-Discriminantes Lineares, que utiliza informações temporais para tentar aproximar a Análise de Discriminantes Lineares em dados não rotulados - como é o caso da DL. Eles testaram o FLsD em condições controladas, onde dados com sobreposição de fala foram removidos, e a característica da base de áudio utilizada é de longos períodos de fala de apenas um locutor (Debates Políticos). Nestas condições, o que foi assumido para o método ser não-supervisionado é válido na grande maioria dos casos. Entretanto, neste trabalho vamos estudar o desempenho do FLsD em situações diferentes - como conversas telefônicas, transmissões de rádio e televisão, e reuniões - e verificar a qualidade do seu resultado. Com base neles, também iremos propor duas modificações para o uso do FLsD. São elas: nova inicialização baseada na saída do segmentador, e o uso simultâneo de múltiplas técnicas de parametrização.

[1] Informações sobre todas as SREs estão disponíveis em `http://www.itl.nist.gov/iad/mig/tests/sre/`.

[2] Informações sobre todas as avaliações estão disponíveis em `http://www.itl.nist.gov/iad/mig/tests/rt/`.

[3] Informações referentes ao ano de 2012.

1.2 Estrutura do Trabalho

Este trabalho está organizado em 5 capítulos. No Capítulo 2, trataremos da arquitetura de um SDL, descrevendo seus principais módulos e algoritmos utilizados, e ao final faremos uma pequena revisão do estado-da-arte. O Capítulo 3 descreve as bases utilizadas nos experimentos, as métricas de desempenho para DL, o sistema de referência, e, por fim, nossa contribuição para reduzir o tempo de processamento na etapa de agrupamento. No Capítulo 4, aplicaremos na tarefa de DL as técnicas de análise e redução de dimensionalidade mencionadas, faremos um estudo sobre o uso de múltiplas técnicas de parametrização do sinal com o uso destas técnicas, e iremos propor modificações a Análise de Semi-Discriminantes Lineares. O Capítulo 5 faz as considerações finais e aponta trabalhos futuros. E por fim, as referências utilizadas ao longo deste trabalho são listadas.

2 DIARIZAÇÃO DE LOCUTOR

Este capítulo descreve o processo de DL e seus principais algoritmos. Iniciaremos mostrando na Seção 2.1 a arquitetura básica de um SDL, situando seus módulos e funções. Após isto, cada um destes módulos será individualmente tratado nas seções 2.2, 2.3, 2.4, 2.5 e 2.6, onde detalharemos alguns dos algoritmos conhecidos da área. E por fim, na Seção 2.7, comentaremos sobre trabalhos e SDLs no estado-da-arte.

2.1 Arquitetura

A DL é um processo onde um conjunto de rótulos, indicando trechos de fala de vários locutores, é encontrado a partir de um sinal de áudio. Para atingir este objetivo, um SDL comumente utiliza 5 módulos (TRANTER; REYNOLDS, 2006; STAFYLAKIS; KATSOUROS, 2011; ANGUERA MIRO et al., 2012), os quais são:

- Parametrização do Sinal: o qual é responsável por extrair informações importantes de um sinal, convertendo-o em *vetores de características*;
- Detecção de Fala: o qual classifica o áudio em trechos de fala e não-fala;
- Segmentação de Locutor: o qual detecta pontos onde existem trocas de locutor, ou seja, pontos que delimitam o fim da fala de um locutor, e o inicio da fala de outro Locutor. Todo trecho entre um ponto e outro é chamado de segmento;
- Agrupamento de Locutor: o qual agrupa os segmentos de acordo com seu respectivo locutor. Idealmente o número de grupos aqui formados é igual ao número de locutores contidos no áudio;
- Re-Segmentação: o qual utiliza os dados dos grupos obtidos pelo módulo de agrupamento para refinar a segmentação. Cada grupo é modelado como um locutor diferente, e a segmentação é refeita para tentar-se obter suas partes semelhantes.

Estes módulos seguem uma ordem sequencial, como mostrado na Figura 2.1, onde a saída de um é também a entrada do outro. Note que os 4 primeiros módulos já são suficientes para se produzir a saída desejada (linha hachurada). Entretanto é comum encontrar SDLs que fazem um processo de iteração entre os módulos de re-segmentação e de agrupamento.

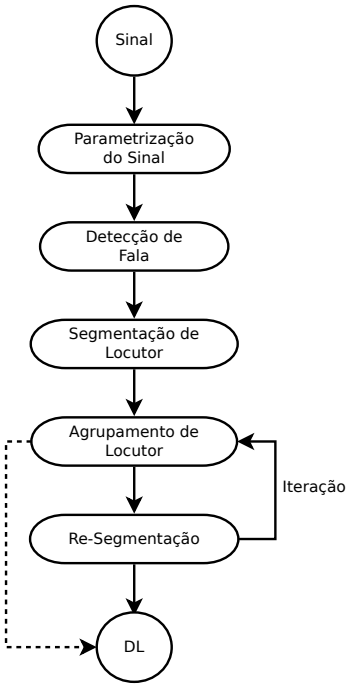


Figura 2.1: Arquitetura de um SDL.

2.2 Parametrização do Sinal

O objetivo do módulo de Parametrização do Sinal é transformar sinais acústicos unidimensionais em vetores de características. Este módulo é comum a diversas áreas do Processamento de Voz, incluindo DL. As técnicas consolidadas datam das décadas de 70 e 80, como o MFCC (*Mel-Frequency Cepstrum Coefficients*) (MERMELSTEIN, 1976), o PLP (*Perceptually Linear Predictive*) (HERMANSKY; HANSON; WAKITA, 1985) e o LPC (*Linear Predictive Coding*) (BUNDY; WALLEN, 1984).

Atualmente, os módulos de parametrização do sinal em SDLs, utilizam alguma destas técnicas citadas, ou variações delas. Muitas vezes, a parametrização também tende a ser específica de domínio, podendo conter sub-módulos adicionais para outros fins (e.g. extração de *beamforming* para dados multicanais, Filtro de Wiener para atenuação de ruído, entre outros) (ANGUERA MIRO et al., 2012).

As técnicas de parametrização podem ser divididas em dois grupos conforme o domínio das características: o Domínio da Frequência e o Domínio do Tempo. A base para as técnicas sobre o Domínio da Frequência é a Análise de Fourier, capaz de transferir um sinal no Domínio do Tempo para o Domínio da Frequência. Já para técnicas no Domínio Tempo, usa-se como base a Predição Linear, onde são estimados coeficientes para prever o comportamento da onda.

A aplicação destas técnicas deve ocorrer sobre ondas periódicas, mas é sabido que as frequências produzidas pelo trato vocal sofrem variações em espaços curtos de tempo. Por esta razão, costuma-se usar janelas do sinal de tamanhos entre 15 e 30 ms, assim podemos dizer que, dentro desta pequena faixa de tempo, o trato vocal está estável e portanto a onda é periódica (PICONE, 1993).

O processo de parametrização estima vetores de características utilizando uma janela deslizante sobre o sinal. A Figura 2.2 nos mostra como atua este módulo. Nela, são extraídos pequenos trechos do sinal de tamanho igual a T_j (janela) a cada T_s segundos (espaçamento). Os valores de T_j e T_s são geralmente entre 10 e 30 ms, sendo que sempre

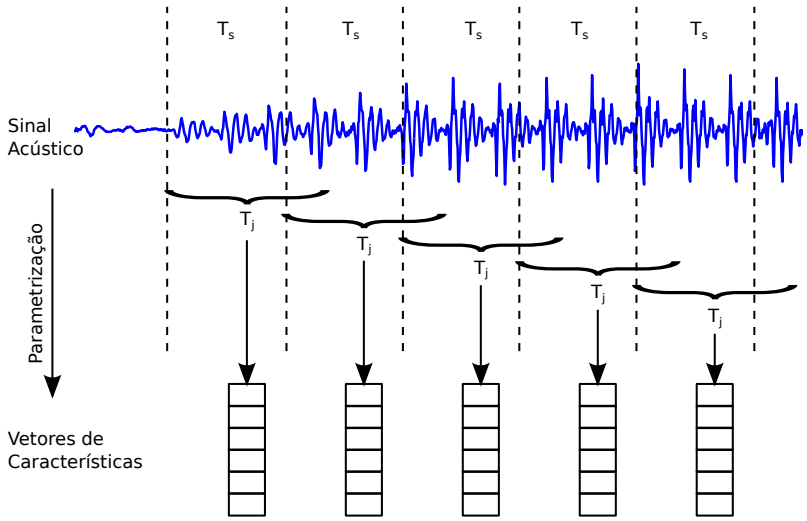


Figura 2.2: Parametrização de um sinal acústico.

$T_j \geq T_s$. Após a aplicação da técnica de parametrização, esta deve retornar um vetor de características. Para DL, é comum encontrar vetores entre 12 e 24 dimensões.

2.2.1 Técnicas Sobre o Domínio da Frequência

As técnicas de parametrização sobre o Domínio da Frequência realizam operações no espectro do sinal (magnitudes de cada frequência), obtido através da Análise de Fourier. Neste trabalho, serão utilizadas duas técnicas neste domínio: os Coeficientes Cepstrais de Frequência Mel (*Mel-Frequency Cepstrum Coefficients* - MFCC) e os Coeficientes de Frequência Mel (*Mel Spectrum Coefficients* - denotado aqui por MEL). Observando o nome já podemos identificar que a diferença entre o MFCC e o MEL é o uso ou não dos coeficientes cepstrais. A Figura 2.3 mostra as etapas para a extração dos vetores de características em ambas as técnicas.

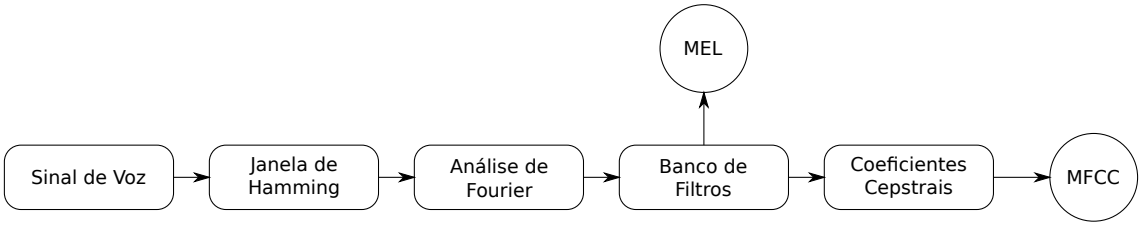


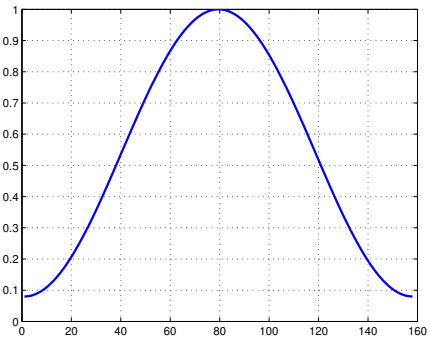
Figura 2.3: Etapas das técnicas de parametrização MFCC e MEL.

A seguir descreveremos detalhadamente cada uma destas etapas (PICONE, 1993):

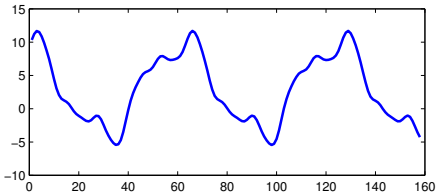
- **Janela de Hamming**: Quando extraímos pequenos trechos de um sinal de voz, dos quais pode-se assumir que são periódicos, é comum que haja descontinuidades nas suas bordas, como é o caso do sinal da Figura 2.4b. Uma vez que a Análise de Fourier assume que o sinal de entrada é periódico, então é necessário uma técnica que seja capaz de suavizar suas bordas sem que haja perda das principais frequências. Neste sentido, usa-se a Janela de Hamming (Figura 2.4a) para atenuar estas descontinuidades do sinal (Figura 2.4c). A janela é equacionada da seguinte forma:

$$\mathbf{s}[n]' = \left(0.54 - 0.46\cos\left(\frac{2\pi(n-1)}{N-1}\right)\right)\mathbf{s}[n] \tag{2.1}$$

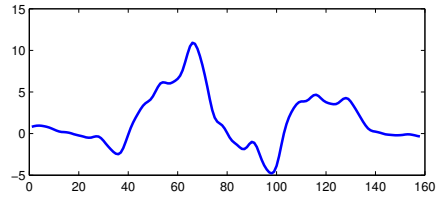
dado um sinal de entrada $\mathbf{s}[n]$ de N amostras.



(a) Janela de Hamming



(b) Sinal sem Janela de Hamming



(c) Sinal com Janela de Hamming

Figura 2.4: Janela de Hamming e sua aplicação em um sinal discretizado.

- **Análise de Fourier**: A Análise de Fourier é utilizada para transferir um sinal no domínio do tempo para o domínio da frequência. Como o áudio digitalizado é uma onda discretizada no tempo, usa-se um caso particular da Análise de Fourier, chamada de Transformada de Fourier de Tempo Discreto (*Discrete-Time Fourier Transform* - DTFT). A DTFT, a qual é definida como:

$$\mathbf{S}(f) = \sum_{n=0}^{N-1} \mathbf{s}[n] e^{-i(2\pi f/f_s)n} \tag{2.2}$$

onde $\mathbf{s}[n]$ é o sinal de entrada, f denota a frequência (em Hertz), f_s a frequência de amostragem do sinal, e N o tamanho do sinal. Como resultado da transformada, $\mathbf{S}$ representa o espectro (energia de cada frequência) do sinal. Os valores de f devem ser menores que $f_s/2$ pois, segundo o Teorema de Nyquist, $\mathbf{S}(f) = 0$ para todos os valores onde $f \geq f_s/2$. Observando a equação, podemos perceber que a energia (ou magnitude) de cada frequência é dada no plano-z. Por isso, para as etapas subsequentes, utiliza-se o espectro de potência, dado por $\mathbf{S}(f)^2$;

- **Escala Mel**: A Escala Mel é uma escala de frequências resultante de dados observados em experimentos psico-acústicos sobre as respostas naturais do sistema auditivo humano (HUANG et al., 2001). Os pesquisadores na época (em 1937) notaram que a frequência percebida pelo ouvido (ou *pitch*) e a frequência real são diferentes, principalmente quando a frequência é superior a 1000 Hz, e elaboraram uma função matemática que mapeia a frequência real (Hertz) para a frequência percebida (Mel), dada por:

$$Mel(f) = 1125ln\left(1 + \frac{f}{700}\right) \tag{2.3}$$

onde f é dado em Hertz. O gráfico na Figura 2.5 mostra o mapeamento das frequências entre 0-16 kHz, para Mel.

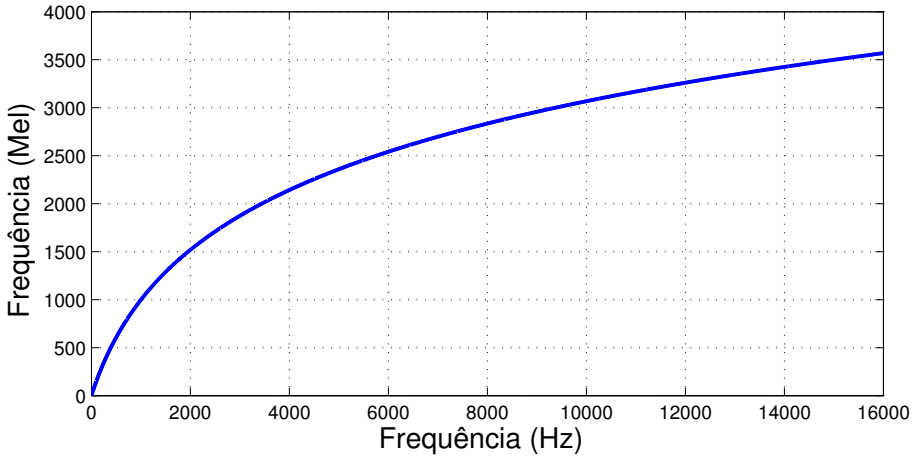


Figura 2.5: Conversão de Hertz para Mel.

- **Análise por Banco de Filtros**: Bancos de Filtros são conjuntos de filtros aplicados a um sinal. No domínio deste trabalho, usa-se bancos de filtros triangulares, aplicados sobre o espectro do sinal, onde o centro de cada filtro é igualmente espaçado segundo a escala mel. Na Figura 2.6 vemos um conjunto de 19 filtros sobre o domínio da frequência limitados entre 300 e 3500 Hz.

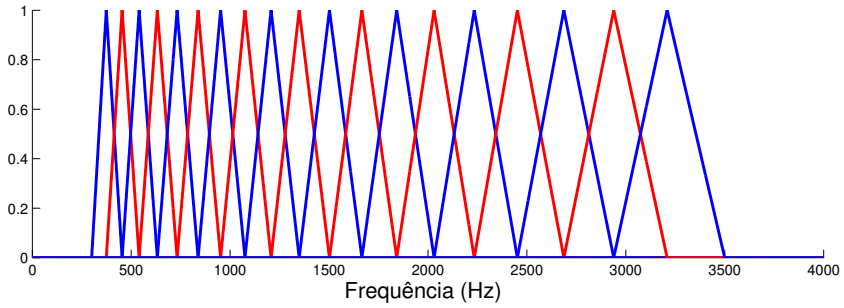


Figura 2.6: Banco de Filtros: 19 filtros triangulares entre 300-3500 Hz, centrados segundo a escala mel.

Podemos gerar coeficientes de energia a partir da soma ponderada de cada filtro sobre a energia das frequências que eles abrangem, da seguinte forma:

$$c_i = \sum_{f=1}^{f_s/2} w_i(f)\mathbf{S}(f) \tag{2.4}$$

no qual i é um número inteiro entre 1 e o número de total filtros, f é a frequência em Hz, $f_s/2$ é a frequência de Nyquist, e w_i é a função que representa o filtro i dado uma frequência f. Conforme pode ser visto na Figura 2.6, a função w_i retorna valores iguais a 0 para frequências que estão fora dos limites do filtro, e menores que 1 para as que estão dentro.

- **Coeficientes Cepstrais**: Os Coeficientes Cepstrais são obtidos através da aplicação da Transformada do Cosseno sobre o logaritmo dos coeficientes oriundos

da Análise por Banco de Filtros, assim:

$$c_i' = \sqrt{\frac{2}{N}} \sum_{j=1}^{N} \log(c_j) \cos\left(\frac{\pi i}{N}(j - 0.5)\right) \tag{2.5}$$

onde c_i' são os coeficientes cepstrais, c_i os coeficiente de energia do banco de filtros, e N o número de filtros.

O principal motivo para a aplicação dos coeficientes cepstrais é o que os filtros do Banco de Filtros sofrem grande sobreposição, fazendo com que os coeficientes sejam fortemente correlacionados. Assim, a aplicação da Transformada do Cosseno descorrelaciona estas energias, permitindo, por exemplo, que sejam usadas matrizes de covariância diagonal em algoritmos de ML.

2.2.2 Técnicas Sobre o Domínio do Tempo

A parametrização do sinal sobre o Domínio do Tempo é baseada na Codificação por Predição Linear (*Linear Predictive Coding* - LPC). O objetivo desta codificação é encontrar um pequeno número de coeficientes capazes de reconstruir um sinal periódico. Isto faz do LPC uma das mais poderosas técnicas para análise de fala, e amplamente utilizada por sistemas de Reconhecimento Automático de Fala. Os principais motivos para isto são (RABINER; JUANG, 1993):

1. Provê um bom modelo do sinal de fala, principalmente nas regiões sonoras, onde o LPC consegue uma boa aproximação do trato vocal;

2. Permite uma estimativa parcimoniosa de características da fala como *pitch*, formantes e espectro;

3. Pode representar o sinal para transmissões de baixa velocidade. Atualmente ele é usado como uma forma de compressão de voz para o padrão GSM;

4. Tem um custo computacional tratável, sendo de simples implementação via software ou hardware;

5. Funciona bem em aplicações de reconhecimento (o que inclui DL, como será visto mais adiante).

Neste trabalho faremos uso de duas técnicas baseadas no LPC: Pares de Linhas Espectrais (*Line Spectral Pairs* - LSP) e Coeficientes Cepstrais de Predição Linear (*Linear Predictive Cepstrum Coeficients* - LPCC).

A Figura 2.7 mostra as etapas necessárias para a obtenção destes coeficientes, que são descritas a seguir:

- **Codificação por Predição Linear**: O LPC baseia-se na idéia de que uma amostra qualquer de um sinal periódico pode ser aproximada por uma combinação linear de suas amostras anteriores. Assim, dado um sinal $\mathbf{s}$, a amostra $\mathbf{s}[n]$ pode ser aproximada por:

$$\hat{\mathbf{s}}[n] \approx \alpha_1 \mathbf{s}[n-1] + \alpha_2 \mathbf{s}[n-2] + ... + \alpha_p \mathbf{s}[n-p] \tag{2.6}$$

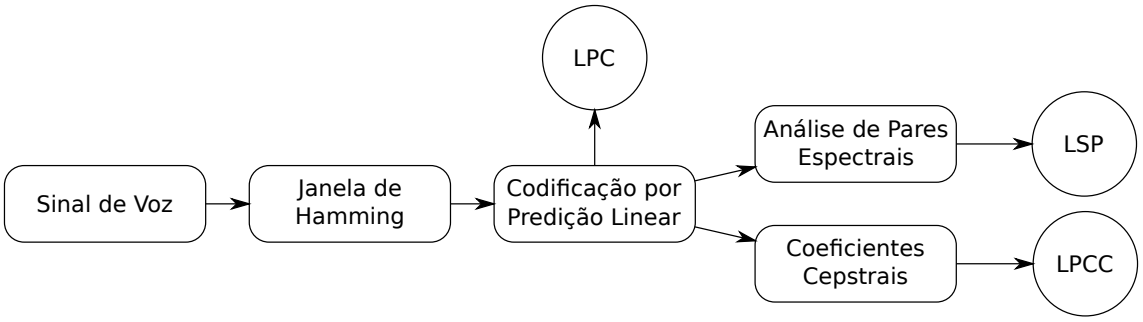


Figura 2.7: Etapas para extração do LPC, LSP e LPCC.

onde $\alpha_{1..p}$ são os coeficientes preditores oriundos do LPC. Podemos converter a equação anterior para uma igualdade, da seguinte forma:

$$\mathbf{s}[n] = \sum_{i=1}^{p} \alpha_i s[n-1] + Gu(n) \tag{2.7}$$

sendo $u(n)$ a excitação normalizada e G o seu ganho. Se expressarmos a Equação 2.7 no domínio-z, podemos obter a relação:

$$S(z) = \sum_{i=1}^{p} \alpha_i z^{-i} S(z) + GU(z) \tag{2.8}$$

que leva a função de transferência:

$$H(z) = \frac{S(z)}{GU(z)} = \frac{1}{1 - \sum_{i=1}^{p} \alpha_i z^{-i}} = \frac{1}{A(z)}. \tag{2.9}$$

Neste sistema, a fonte e excitação $Gu(z)$ servirá como entrada para o sistema todo-polo, $H(z) = 1/A(z)$, produzir o sinal de fala $\mathbf{s}[n]$.

Como a Equação 2.6 é uma aproximação, podemos medir seu erro quadrático por:

$$e_m = (\mathbf{s}[n-1] - \hat{\mathbf{s}}[n])^2 \tag{2.10}$$

e encontrar os coeficientes α pela minimização desta equação. Na literatura encontramos vários métodos para derivar estes coeficientes (RABINER; SCHAFER, 1979).

A representação dos coeficientes LPC no Domínio da Frequência pode ser vista observando seu Envelope Espectral[1]. Na Figura 2.8 temos o logaritmo do espectro de um sinal de voz[2] (linha azul), e seus respetivos Envelopes Espectrais (linhas pretas) para o LPC com 5, 10 e 20 coeficientes. Note que, conforme aumentamos o número de coeficientes, o envelope melhor se adapta ao espectro original.

[1] O Envelope Espectral do LPC é calculado por $|H(e^{i\pi(f/f_s)})|$.

[2] 30 ms e Janela de Hamming

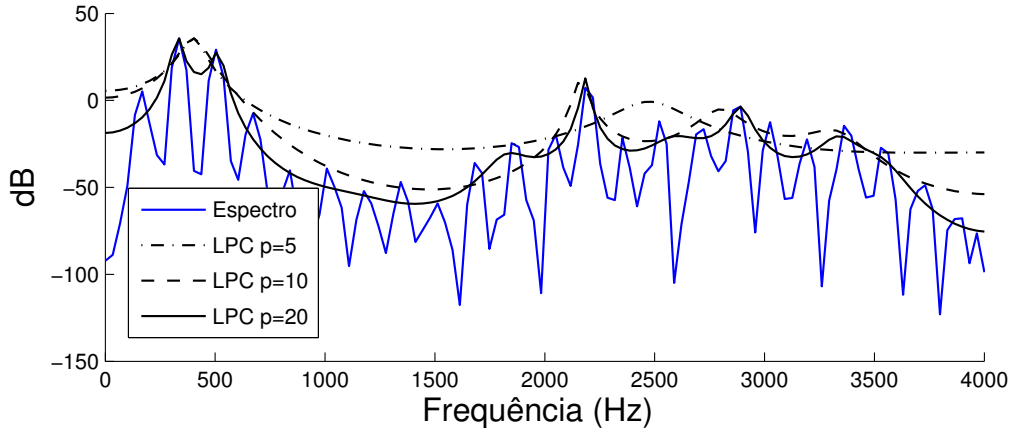


Figura 2.8: Energia do espectro (em dB) e Envelope Espectral do LPC com $p = \{5, 10, 20\}$.

- **Pares de Linhas Espectrais**: O LSP é uma técnica aplicada sobre o LPC capaz de melhorar a quantização e interpolação do sinal. Dado $A(z) = 1 - \sum_{i=1}^{p} \alpha_i z^{-i}$ (denominador da Equação 2.9), pode-se expressa-lo como:

$$A(z) = \frac{1}{2}[P(z) + Q(z)] \tag{2.11}$$

onde $P(z)$ é um polinômio palíndromo e $Q(z)$ um polinômio anti-palíndromo:

$$P(z) = A(z) + z^{-(p+1)}A(z^{-1}) \quad \text{e} \quad Q(z) = A(z) - z^{-(p+1)}A(z^{-1}). \tag{2.12}$$

As raízes de $P(z)$ e $Q(z)$ cujos ângulos no plano-z forem menores do que π, representam os coeficientes LSP. Valores maiores do que π são espelhados no quadrantes onde a parte imaginária é negativa, e por isso são ignorados. Para ilustrar, a Figura 2.9 mostra, no plano-z, todas as 10 raízes dos polinômios $A(z)$, $P(z)$ e $Q(z)$. Neste exemplo, o sinal analisado é o mesmo do item anterior (Figura 2.8). Note que as raízes de $P(z)$ e $Q(z)$ estão nas bordas do círculo unitário, e que a distância entre cada par de raízes (p, q) é inversamente proporcional à distância que a raiz a mais próxima está da origem.

- **Coeficientes Cepstrais**: É possível ainda extrair coeficientes cepstrais a partir dos coeficientes do LPC, pois o filtro de predição linear é estável (PICONE, 1993). Eles são obtidos por:

$$c(i) = \begin{cases} c(i) = -\alpha_i & \text{se } i = 1 \\ c(i) = -\alpha_i - \sum_{j=1}^{i-1} \left(1 - \frac{j}{i}\right) \alpha_i c_{i-j} & \text{se } i = 2, ..., p. \end{cases} \tag{2.13}$$

2.3 Detecção de Fala

O objetivo da Detecção de Fala é classificar as partes de um áudio em fala e não-fala, como mostra a Figura 2.10. Por não-fala podemos considerar trechos de silêncio, ruídos

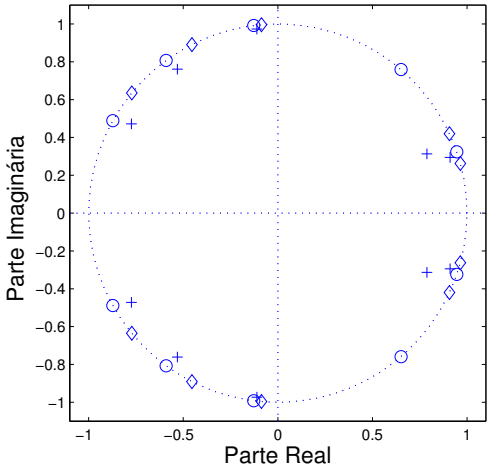


Figura 2.9: Raízes de $P(z)$ (diamante), $Q(z)$ (círculo) e $A(z)$ (cruz).

de canal e ruídos de ambiente (e.g. eletrodomésticos ligados, ventanias, música, entre outros). Sua importância para DL é significante em dois aspectos (ANGUERA MIRO et al., 2012): (1) a métrica utilizada para medir o desempenho de sistemas de DL considera a quantidade de fala perdida e a quantidade de não-fala classificada erroneamente como fala; (2) trabalhar apenas com trechos de fala leva ao treinamento de modelos acústicos mais precisos, melhorando o desempenho do sistema como um todo.

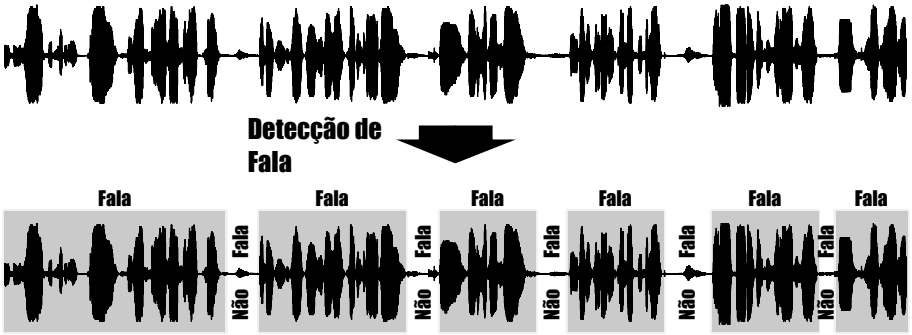


Figura 2.10: Saída de um módulo de detecção de fala, que classifica trechos do áudio em fala e não-fala.

O módulo de Detecção de Fala é utilizado não apenas na DL, mas também em outras áreas do Processamento de Voz, como Reconhecimento de Fala e Reconhecimento de Linguagem (LI; MA; LEE, 2013). Na literatura é comummente chamado de *Speech Activity Detection* (SAD) ou *Voice Activity Detection* (VAD). Neste trabalho adotaremos a sigla VAD.

Historicamente, no inicio os VADs não compunham um módulo separado na DL, e a detecção de não-fala era apenas um produto dos módulos de segmentação e agrupamento. Desta forma, ao final da DL, deveria-se idealmente restar um *cluster* contendo apenas dados de não-fala (WILCOX et al., 1994). Com o tempo, se tornou evidente que estas abordagens eram inferiores quando comparadas a outras que utilizavam um VAD dedicado.

Atualmente encontramos dois tipos de abordagens para VADs:

- **Abordagens baseadas em modelo**: utiliza-se técnicas de Aprendizagem de Máquina

para classificar entre fala e não-fala. É comum encontrar Modelos Ocultos de Markov (*Hidden Markov Models* - HMM) de 2 ou 3 estados (WOOTERS; HUIJBREGTS, 2008; FREDOUILLE; EVANS, 2008; LEEUWEN; KONEčNý, 2008), sendo um estado de fala, um estado de não-fala, e um terceiro estado - utilizado por alguns SDLs - de silêncio. Neste caso a classificação é realizada pelo Algoritmo de Viterbi. Também podemos encontrar na literatura outros trabalhos que utilizam Máquinas de Vetores de Suporte (*Support Vector Machine* - SVM) (TEMKO; MACHO; NADEU, 2007; RAMíREZ et al., 2006). Estas abordagens baseadas em modelo tem sido fortemente utilizadas, pois apresentam resultados superiores. Entretanto, elas dependem de dados externos, ficando assim sujeitas a degradações de desempenho quando há variações nas condições acústicas (como variações de canal ou de ambiente), e desta forma, mais dependentes do domínio.

- **Abordagens não-baseadas em modelo**: podem utilizar diferentes estratégias para inferir regiões de fala de não-fala, como por exemplo (RAMIREZ; GÓRRIZ; SEGURA, 2007; SHEN; HUNG; LEE, 1998): limiares de decisão sobre o valor da energia, detecção da frequência fundamental (*pitch*), taxa de cruzamento em zero, medidas de periodicidade, entropia espectral, entre outros. Apesar de possuírem um desempenho inferior às abordagens baseadas em modelo, estas normalmente são aplicáveis a qualquer domínio.

2.4 Segmentação de Locutor

O módulo de Segmentação de Locutor tem o objetivo de identificar os pontos em um áudio onde ocorrem mudanças (ou trocas) de locutor. A Figura 2.11 ilustra este processo. Como esta é uma etapa subsequente ao VAD, as partes de não-fala são removidas antes da sua aplicação.

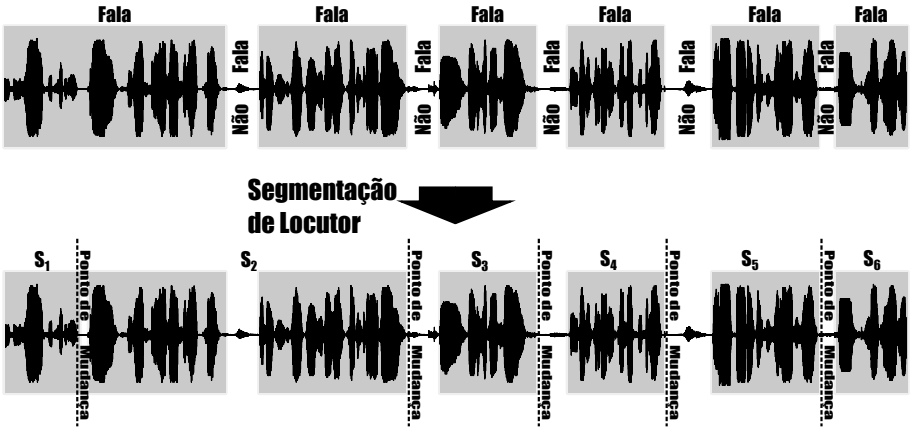


Figura 2.11: A saída do módulo de segmentação encontra os prováveis pontos de troca de locutor.

2.4.1 Algoritmo de Janelas Crescentes

Apresentado por Chen e Gopalakrishnan (CHEN; GOPALAKRISHNAN, 1998), este algoritmo propõe o uso de uma janela crescente, que expande-se ao longo do áudio até que um ponto de troca seja encontrado. Durante cada expansão, tenta-se encontrar um ponto de troca de locutor dentro da própria janela, dividindo-a em duas janelas menores e

adjacentes. Quando um ponto de troca é encontrado, então a janela volta ao seu tamanho inicial e seu primeiro índice passa a ser o ponto de troca.

O Algoritmo 1 mostra o Algoritmo de Janelas Crescentes completo. Note a partir do laço na linha 7, que o algoritmo percorre toda a janela principal, dividindo-a em duas janelas adjacentes de tamanho variável. O ponto p, que separa duas janelas onde a distância d entre elas for a maior dentre todas as outras e também maior do que o limiar β (linhas 5 e 9), é tido como ponto de mudança (linha 15). Quando isto acontece, o inicio da janela principal passa a ser o ponto p (linha 16) e seu tamanho volta a ser j (linha 17). Entretanto, se $d \leq \beta$ para todas as janelas adjacentes, então incrementa-se em e o tamanho da janela principal (linha 20) e recomeça o laço.

Entrada: $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$: Vetores de Característica
Entrada: j : Tamanho inicial da janela
Entrada: e : Tamanho de cada expansão da janela
Entrada: m : Tamanho mínimo de uma janela
Saída: β : Limiar
Saída: S : Conjunto de pontos de troca de locutor

```
1  a ← 1; // Início da janela
2  c ← j; // Fim da janela
3  S ← ∅;
4  enquanto c < N faça
       /* Busca por ponto de mudança no intervalo [x_a, ..., x_c]
          */
5      max_d ← β;
6      p ← 0;
7      para cada b ∈ {m, m+1, m+2, ..., c−a} faça
8          d ← distância([x_a, ..., x_{a+b}], [x_{a+b+1}, ..., x_c]);
9          se d > max_d então
10             max_d = d;
11             p = a + b;
12         fim
13     fim
       /* Adiciona o ponto p (se encontrado) ao conjunto de
          pontos de mudança                                    */
14     se p então
15         S = S ⋃ {p};
16         a ← p;
17         c ← p + j;
18     fim
19     senão
20         c ← c + e; // Incrementa tamanho da janela
21     fim
22 fim
```

Algoritmo 1: Algoritmo de Janelas Crescentes

Este algoritmo tem a desvantagem de ser computacionalmente custoso, pois grandes trechos de fala de um mesmo locutor podem acarretar excessivos cálculos de distância à

medida que a janela principal cresce. Em contra partida, as sucessivas expansões auxiliam na criação de melhores modelos estatísticos, que normalmente são a base das medidas de distância.

2.4.2 Algoritmo de Janelas Deslizantes

O algoritmo de Janelas Deslizantes é composto de duas etapas. A primeira etapa percorre os vetores de características do áudio, calculando a distância entre duas janelas adjacentes de tamanho fixo. Na segunda etapa, o algoritmo busca por máximos locais nas distâncias encontradas. Um máximo local, cuja diferença entre ele e seus dois mínimos locais à direita e à esquerda, for maior que um dado limiar, é considerado um ponto de mudança. Na formulação encontrada em (DELACOURT; KRYZE; WELLEKENS, 2000) os autores usam como limiar o desvio padrão das distâncias, multiplicado por algum fator.

Ao computarmos sequencialmente as distâncias entre duas janelas adjacentes, teremos como resultado algo semelhante a Figura 2.12, que mostra estas distâncias (linha preta) computadas sobre janelas de tamanho w. Quando observamos os valores de d ao longo do áudio, podemos notar alguns picos (ou máximos locais), que podem representar pontos de mudança de locutor. Entretanto nem todos determinam verdadeiros pontos de mudança. A decisão sobre qual ponto é relevante ou não cabe ao limiar escolhido. Normalmente picos mais agudos tem maiores chances de representar pontos de mudança.

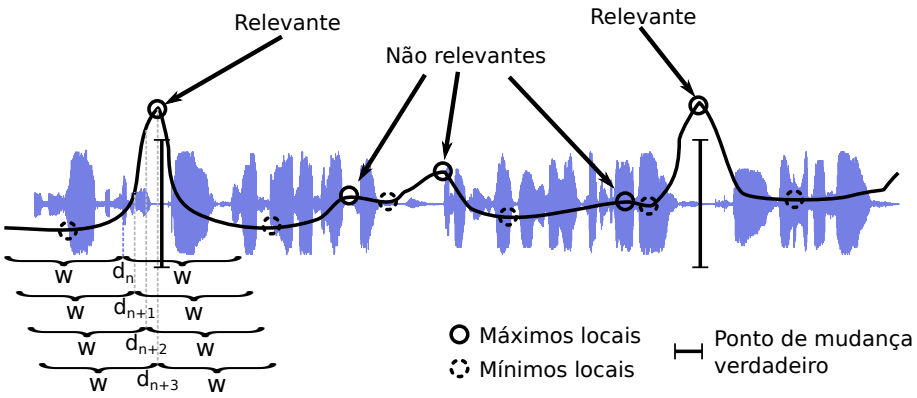


Figura 2.12: Algorítmo de Janelas Deslizantes.

O Algoritmo 2 mostra detalhadamente o Algoritmo de Janelas Deslizantes. A primeira etapa onde são computadas as distâncias, encontra-se no laço na linha 5. O limiar de decisão, calculado pelo desvio padrão das distância multiplicado por um fator α, é mostrado na linha 12. Note também, na linha 19, que ambos os mínimos locais (a direita e a esquerda) devem ser menores o suficiente que o máximo local para que seja satisfeita a condição. Comparando com a Figura 2.12, note que valores baixos de β tendem a super-segmentar o áudio (i.e. dividi-lo excessivamente em muitos segmentos pequenos). Já valores muito altos tendem a gerar segmentos com mais de um locutor.

2.4.3 Algoritmo DistBIC

O Algoritmo DistBIC foi apresentado por Delacourt e Wellekens (DELACOURT; KRYZE; WELLEKENS, 2000). Ele consiste em uma técnica de segmentação de duas etapas. A primeira realiza uma super-segmentação no áudio, e a segunda complementa a etapa anterior juntando segmentos pequenos adjacentes com maior chance (segundo o critério ΔBIC) de pertencerem ao mesmo locutor.

Entrada: $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$: Vetores de Característica
Entrada: w : Tamanho inicial da janela
Entrada: s : Tamanho do deslocamento
Entrada: α : Fator para o limiar de decisão
Saída: S : Conjunto de pontos de troca de locutor

$S \leftarrow \emptyset$;
$\mathbf{w}^{(d)} \leftarrow [1, 2, ..., w]$; `// Índices da janela à direita`
$\mathbf{w}^{(e)} \leftarrow [w+1, w+2, ..., 2j]$; `// Índices da janela à esquerda`
`// Computa vetor de distâncias ao longo dos dados`
; $i = 1$;
enquanto $\boldsymbol{w}_j^{(e)} < N$ **faça**
    `// Calcula a distância entre as janelas adjacentes`
    $\mathbf{d}_i \leftarrow$ distância$([\mathbf{x}_{\mathbf{w}_1^{(d)}}, ..., \mathbf{x}_{\mathbf{w}_j^{(d)}}], [\mathbf{x}_{\mathbf{w}_1^{(e)}}, ..., \mathbf{x}_{\mathbf{w}_j^{(e)}}])$;
    `// Armazena referência para o centro da janela`
    $\mathbf{p}_i \leftarrow \mathbf{w}_j^{(d)}$;
    `// Incrementa índices das janelas`
    $\mathbf{w}^{(d)} \leftarrow \mathbf{w}^{(d)} + s$;
    $\mathbf{w}^{(e)} \leftarrow \mathbf{w}^{(e)} + s$;
    $i \leftarrow i + 1$;
fim
$\beta \leftarrow$ desvio_padrão$(\mathbf{d}) * \alpha$; `// Limiar de decisão`
$M \leftarrow$ índices_de_máximos_locais$(\mathbf{d})$;
para cada $m \in M$ **faça**
    $ml_d \leftarrow$ mínimo_local_a_direita$(\mathbf{d}_m)$;
    $ml_e \leftarrow$ mínimo_local_a_esquerda$(\mathbf{d}_m)$;
    `// Diferenças entre mínimos e máximos locais`
    $\delta_d \leftarrow d[m] - ml_d$;
    $\delta_e \leftarrow d[m] - ml_e$;
    se $\beta < \min(\delta_d, \delta_e)$ **então**
        $S = S \bigcup \{p[m]\}$;
    fim
fim

Algoritmo 2: Algoritmo de Janelas Deslizantes

Na primeira etapa, uma super-segmentação é realizada com o algoritmo de Janelas Deslizantes, configurado com um α baixo. Na segunda etapa, ele realiza um refinamento dos "candidatos" a pontos de mudança, encontrados na primeira etapa. Para isso, todos os segmentos são comparados com seus segmentos adjacentes através de alguma medida de distância, mas com um limiar maior. Quando o valor calculado é menor do que este limiar, então os segmentos são unidos. Como resultado, obtém-se segmentos maiores, o que é benéfico para algoritmos de agrupamento.

2.5 Agrupamento de Locutor

O objetivo deste módulo é encontrar segmentos semelhantes, oriundos do módulo de Segmentação de Locutor, e agrupá-los de acordo com os respectivos locutores. Idealmente, o número final de grupos corresponderá ao número de falantes presentes no áudio. A Figura 2.13 mostra a entrada (segmentos) e saída (grupos) deste módulo.

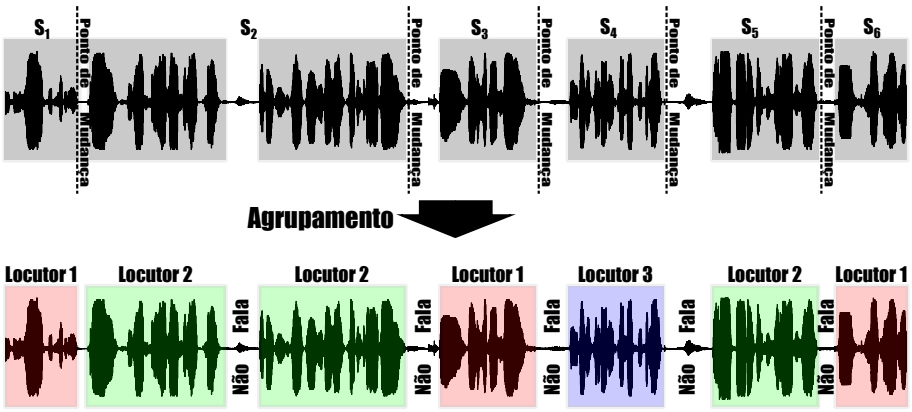


Figura 2.13: Saída do Módulo de Agrupamento: Segmentos agrupados de acordo com o seu locutor.

Atualmente, os SDLs utilizam algum tipo de abordagem Hierárquica Aglomerativa ou Divisiva (TANG et al., 2012). Sendo que o funcionamento de ambas ocorre de maneira oposta. A abordagem aglomerativa (também chamada de *Bottom-Up*), durante a inicialização, considera todos os segmentos como *clusters*, e, se não houver algum critério de parada, vai juntando-os iterativamente durante sua execução, até que se forme um único grande *cluster*. Na Seção 2.5.1 trataremos detalhadamente desta abordagem. Já a abordagem divisiva (também chamada de *Top-Down*) segue o caminho oposto. Inicialmente têm-se um único *cluster*, que sofre sucessivas divisões ao longo da execução, até que o critério de parada seja encontrado, ou até que os dados estejam totalmente fragmentados. Detalhes desta última abordagem não serão apresentados neste trabalho devido ao seu pouco uso na tarefa de DL.

2.5.1 Agrupamento Hierárquico Aglomerativo

O Agrupamento Hierárquico Aglomerativo (*Agglomerative Hierarchical Clustering* - AHC) é uma abordagem de agrupamento que, a cada iteração, une (ou aglomera) o par de *clusters* considerados mais próximos. A inicialização do AHC toma cada entidade inicial (ou segmento de fala, no caso sistemas de DL) como um *cluster*, onde a distância entre todo par de *clusters*, c_i e c_j, é calculada e armazenada em uma matriz, denominada Matriz de Dissimilaridade, $\mathbf{D}$, sendo $\mathbf{D}_{ij} = d(c_i, c_j)$. Dado um conjunto de *clusters* C, onde $|C| = N$, os principais passos do AHC são (MIRKIN, 1996):

1. Estimar matriz $\mathbf{D}_{ij} = d(c_i, c_j), \forall i, j \in 1, ..., N$.
2. Encontrar o menor valor $\mathbf{D}_{ij}$, onde $i \neq j$;
3. Juntar c_i e c_j em um novo *cluster* $c_{i'}$;
4. Transformar $\mathbf{D}$, e calcular para a linha i as novas dissimilaridades entre $c_{i'}$ e os demais *clusters*, e remover a linha e coluna j;

5. Verificar critério de parada. Caso negativo, voltar ao passo 2.

A complexidade do AHC no pior caso é $O(n^2)$. Isto acontece quando não há nenhum critério de parada, e assim as uniões ocorrem sucessivamente até restar apenas 1 *cluster*.

No passo 4, as novas dissimilaridades podem ser calculadas diretamente através de alguma medida de distância d, ou por meio de algum critério de ligação. Na literatura encontramos 3 critérios comuns de ligação (HASTIE; TIBSHIRANI; FRIEDMAN, 2001):

- Distância Máxima: usa-se a maior distância entre os *clusters* i e j para com outro *cluster* k:
$$\mathbf{D}_{ik} = \max\{\mathbf{D}_{ik}, \mathbf{D}_{jk}\} \tag{2.14}$$
- Distância Mínima: usa-se a menor distância entre os *clusters* i e j para com outro *cluster* k:
$$\mathbf{D}_{ik} = \min\{\mathbf{D}_{ik}, \mathbf{D}_{jk}\} \tag{2.15}$$
- Distância Média: usa-se a média das distâncias entre os os *clusters* i e j para com outro *cluster* k:
$$\mathbf{D}_{ik} = \frac{1}{2}(\mathbf{D}_{ik} + \mathbf{D}_{jk}) \tag{2.16}$$

No AHC, o critério de parada é positivo apenas quando todas as distâncias na matriz de dissimilaridade forem acima de um dado limiar. Desta forma, supondo um limiar de valor β, deve-se, à cada iteração do AHC, verificar se $\mathbf{D}_{ij} > \beta$ para todo $i \neq j$. Quando isto ocorrer, a execução é parada e retorna-se os *clusters* encontrados.

2.5.2 Medidas de Distância

Tanto os algoritmos de Segmentação de Locutor quanto o de Agrupamento de Locutor mostrados, utilizam alguma medida de distância (denotada pela função $d()$) entre conjuntos de dados. É comum o emprego de medidas estatísticas que meçam a dissimilaridade entre as distribuições de probabilidade de conjuntos de dados. Nesta seção trataremos de cinco medidas de distância frequentemente utilizadas por algoritmos em DL (ANGUERA MIRO et al., 2012). Todas elas de cunho estatístico.

2.5.2.1 ΔBIC

O ΔBIC é certamente a medida de distância mais utilizada em DL (ANGUERA MIRO et al., 2012). Seu calculo é baseado no Critério de Informação Bayesiano (*Bayesian Information Criterion* - BIC) para seleção de modelos, dado por:

$$BIC(M, \mathbf{X}) = \log \mathcal{L}(\mathbf{X}, M) - \lambda \frac{1}{2} \#(M) \log(N) \tag{2.17}$$

onde $\mathcal{L}$ é a verossimilhança do conjunto de dados $\mathbf{X}$ (de tamanho N) para um modelo estatístico M, λ é um fator de peso e $\#(M)$ o número de parâmetros do modelo M.

Para interpretar a Equação 2.17, primeiro devemos observar a verossimilhança. Sabemos que quanto maior for seu valor, melhor será o modelo M para representar o conjunto de dados $\mathbf{X}$. Esta é a primeira parte da equação. A segunda parte constitui um fator de penalidade proporcional a complexidade do modelo M (seu número de parâmetros) e ao tamanho do conjunto de dados $\mathbf{X}$. Então, podemos interpretar o valor BIC da mesma

forma que a verossimilhança. Ou seja, quanto maior, melhor o modelo. Entretanto, dada a penalidade, nem sempre o modelo com maior verossimilhança será o mais adequado.

Dado 3 conjuntos de dados, representados por $\mathbf{X}_i$, $\mathbf{X}_j$ e $\mathbf{X}_z$ (onde $\mathbf{X}_z$ é a simples concatenação de $\mathbf{X}_i$ e $\mathbf{X}_j$) e seus respectivos modelos por M_i,M_j e M_z. Iniciaremos a definição da distância ΔBIC formulando duas hipóteses:

- H_0: O modelo M_z melhor representa os dados se $BIC(M_z, \mathbf{X}_z) \geq BIC(M_i, \mathbf{X}_i) + BIC(M_j, \mathbf{X}_j)$.
- H_1: Os modelos M_i e M_j melhor representam os dados se $BIC(M_i, \mathbf{X}_i) + BIC(M_j, \mathbf{X}_j) > BIC(M_z, \mathbf{X}_z)$.

Em outras palavras, quando a hipótese H_0 é verdadeira, então assume-se que ambos os conjuntos pertencem a mesma distribuição de probabilidades. Caso contrário, se H_1 for verdadeira, então os segmentos pertencem a distribuições diferentes. Em termos de DL, H_0 diz que os conjuntos de dados pertencem ao mesmo locutor, e H_1 diz que são de locutores diferentes. Se rearranjarmos as inequações em relação a 0 e multiplicarmos ambas por (-1), teremos as seguintes condições para as hipóteses:

$$H_0 : \text{ se } BIC(M_z, \mathbf{X}_z) - BIC(M_i, \mathbf{X}_i) - BIC(M_j, \mathbf{X}_j) \leq 0 \tag{2.18}$$

$$H_1 : \text{ se } BIC(M_z, \mathbf{X}_z) - BIC(M_i, \mathbf{X}_i) - BIC(M_j, \mathbf{X}_j) > 0 \tag{2.19}$$

Assim, definimos a distância ΔBIC como:

$$\Delta\text{BIC}(\mathbf{X}_i, \mathbf{X}_j) = BIC(M_z, \mathbf{X}_z) - BIC(M_i, \mathbf{X}_i) - BIC(M_j, \mathbf{X}_j) \tag{2.20}$$

onde H_0 ocorre quando $\Delta\text{BIC} \leq 0$ e H_1 quando $\Delta\text{BIC} > 0$ (equações 2.18 e 2.19).

Se considerarmos M como um processo Gaussiano multivariado, ou seja, $M = \mathcal{N}(\mu, \mathbf{\Sigma})$, podemos encontrar a forma fechada da equação 2.20:

$$\Delta\text{BIC}(\mathbf{X}_i, \mathbf{X}_j) = N_z \log |\mathbf{\Sigma}_z| - N_i \log |\mathbf{\Sigma}_i| - N_j \log |\mathbf{\Sigma_j}| - \lambda P \tag{2.21}$$

em que $P = \frac{1}{2}(d - \frac{1}{2}d(d+1)) \log N_z$, e d é a dimensionalidade dos dados.

O ΔBIC é amplamente utilizado na DL como critério de parada, cujo limiar é configurado com o valor 0. Assim, o AHC pára sua execução quando houver um valor positivo na matriz de dissimilaridade.

2.5.2.2 GLR

O *Generalized Likelihood Ratio* (GLR) é uma métrica baseada também em verossimilhança e resultante da relação entre duas hipóteses (GISH; SCHMIDT, 1994; MIRO, 2006). De maneira semelhante ao ΔBIC, temos as hipóteses H_0 e H_1, onde H_0 considera que os dois segmentos analisados pertencem ao mesmo locutor, e H_1 os considera de diferentes locutores. A relação $\frac{H_0}{H_1}$ entre a verossimilhança destas hipóteses, é dada por:

$$GLR(\mathbf{X}_i, \mathbf{X}_j) = \frac{H_0}{H_1} = \frac{\mathcal{L}(\mathbf{X}_z, M_z)}{\mathcal{L}(\mathbf{X}_i, M_i)\mathcal{L}(\mathbf{X}_j, M_j)} \tag{2.22}$$

e a distância GLR determinada por $d_{GLR}(\mathbf{X}_i, \mathbf{X}_j) = -\log(GLR(\mathbf{X}_i, \mathbf{X}_j))$.

Em Gish *et al.* (GISH; SIU; ROHLICEK, 1991), o autor reescreve a equação 2.22 como sendo o produto de duas relações de verossimilhança:

$$GLR = GLR_\mu GLR_\Sigma \tag{2.23}$$

onde GLR_μ e GLR_Σ são as relações de verossimilhança relacionadas à média e à covariância, respectivamente. A forma fechada do GLR_Σ é dada por:

$$GLR_\Sigma(\mathbf{X}_i, \mathbf{X}_j) = N_z \log|\Sigma_z| - N_i \log|\Sigma_i| - N_j \log|\Sigma_j|. \tag{2.24}$$

Note que a equação 2.24 é a igual a equação 2.21 quando $\lambda = 0$, ou seja, quando não há penalidade. Futuramente neste trabalho apresentaremos os resultados de DL para d_{GLR} e d_{GLR_Σ}, onde iremos constatar que o uso da média na tarefa de agrupamento de locutor é insignificante.

2.5.2.3 KL2

A distância KL2 é uma simetrização da divergência de Kullback-Leibler (KL) (SIEGLER et al., 1997). A divergência KL é dada por:

$$KL(\mathbf{X}_i, \mathbf{X}_j) = \sum_{\mathbf{x}\in\mathbf{X}_i} \log\left(\frac{P_i(\mathbf{x})}{P_j(\mathbf{x})}\right) P_i(\mathbf{x}) \tag{2.25}$$

onde P_i e P_j são as f.d.p. de $\mathbf{X}_i$ e $\mathbf{X}_j$ respectivamente. Ao observarmos esta equação, podemos notar que ela não é simétrica em relação a seus parâmetros, ou seja, existem casos onde $KL(\mathbf{X}_i, \mathbf{X}_j) \neq KL(\mathbf{X}_j, \mathbf{X}_i)$. Visto que a definição de distância exige simetria, Siegler *et al.* (SIEGLER et al., 1997) apresentaram a distância KL2 como sendo a simples soma de duas divergências KL com parâmetros trocados:

$$d_{KL2}(\mathbf{X}_i, \mathbf{X}_j) = KL(\mathbf{X}_i, \mathbf{X}_j) + KL(\mathbf{X}_j, \mathbf{X}_i). \tag{2.26}$$

Podemos obter a forma fechada da divergência KL considerando as f.d.p. como processos Gaussianos multivariados (MIRO, 2006):

$$KL(\mathbf{X}_i, \mathbf{X}_j) = \frac{1}{2}tr[(\Sigma_i - \Sigma_j)(\Sigma_i^{-1} - \Sigma_j^{-1})] + \frac{1}{2}tr[(\Sigma_i^{-1} - \Sigma_j^{-1})(\mu_i - \mu_j)(\mu_i - \mu_j)^T]. \tag{2.27}$$

e substituindo 2.27 em 2.26, temos a forma fechada de d_{KL2}.

2.5.2.4 ICR

O *Information Change Rate* (ICR), apresentado por Han e Narayanam em 2007 (HAN; NARAYANAN, 2007), é uma medida de distância fundamentada na Teoria da Informação, que mede o quanto de informação (entropia) foi alterado ao se unir dois segmentos (ou *clusters*).

Primeiramente vamos considerar $H(\mathbf{X}) = \sum_{\mathbf{x}\in\mathbf{X}} p(\mathbf{x}) \log p(\mathbf{x})$ como a entropia de $\mathbf{X}$. O ICR então será:

$$d_{ICR}(\mathbf{X}_i, \mathbf{X}_j) = H(\mathbf{X}_z) - \frac{N_i H(\mathbf{X}_i) + N_j H(\mathbf{X}_j)}{N_i + N_j}. \tag{2.28}$$

Outra forma de calcular o ICR é através da normalização do GLR:

$$d_{ICR}(\mathbf{X}_i, \mathbf{X}_j) = \frac{d_{GLR}(\mathbf{X}_i, \mathbf{X}_j)}{N_i + N_j}. \tag{2.29}$$

2.5.2.5 Bhattacharyya

A Distância de Bhattacharyya é uma das mais famosas distâncias estatísticas encontradas na literatura. Entretanto é raramente empregada em DL. Seu principal uso está relacionado a aplicações de Identificação e Verificação de Locutor (CAMPBELL J.P., 1997). Mesmo assim, optamos por descreve-la neste trabalho, pois será utilizada em experimentos futuros a título de comparação.

A Distância de Bhattacharyya, quando calculada sobre distribuições Gaussianas, nos dá o limite superior do erro de Bayes[3] de classificação (FUKUNAGA, 1990). A sua forma fechada é dada por:

$$d_{BAT} = \frac{1}{8}(\mu_i - \mu_j)^T \left(\frac{\Sigma_i + \Sigma_j}{2}\right)^{-1} (\mu_i - \mu_j) + \frac{1}{2}\log \frac{\left|\frac{\Sigma_i + \Sigma_j}{2}\right|}{\sqrt{|\Sigma_i||\Sigma_j|}} \tag{2.30}$$

Uma extensão desta distância para GMMs pode ser encontrada em (YOU; LEE; LI, 2009), onde também é aplicada ao problema de Reconhecimento de Locutor.

2.6 Re-Segmentação

O objetivo do módulo de re-segmentação é realizar uma nova e mais precisa segmentação no áudio, aproveitando-se de informações previamente obtidas sobre seus locutores. Como a tarefa de DL não permite que exista nenhuma informação *a priori*, é utilizado então a saída do módulo de agrupamento de locutor. Em condições normais, espera-se que esta saída contenha conjuntos de segmentos ajuntados por locutor. Assim, o algoritmo de Re-Segmentação poderá usufruir de maiores quantidades de dados para tentar estimar os pontos de mudança.

A re-segmentação é sempre realizada pelo Algoritmo de Viterbi. Este é um algoritmo de Programação Dinâmica aplicado no contexto de HMMs, que busca pela sequência mais provável de estados que resulte nos dados observados. Para sua aplicação, então, é necessário que haja um HMM previamente treinado - i.e. com sua matriz de probabilidades de transição entre estados já definida - e as probabilidades dos dados observados para cada estado (RABINER, 1989).

Quando adaptamos para o problema de DL, consideramos cada estado do HMM como um modelo estatístico (e.g. uma GMM) que represente um locutor. E como dados observados, as probabilidades inferidas por estes modelos para todos os vetores de características. As probabilidades de transição partindo de um estado para outro devem ser iguais, assim como as probabilidades de transição para o mesmo estado. Desta forma, dada uma matriz de transição $\mathbf{A}$, temos que $\mathbf{A}_{ij} = p$ quando $i \neq j$, e $\mathbf{A}_{ij} = q$ quando $i = j$. Normalmente usa-se $p < q$. Esta é uma decisão lógica pois, (temporalmente) em uma conversa, os locutores alternam-se em períodos maiores que 1 segundo. Dado que a parametrização do sinal normalmente ocorre a cada 10 ou 20 ms, esta configuração assegura que trocas de estado ocorram com menor frequência.

[3] O erro de Bayes é o menor erro de classificação possível que pode ser obtido de duas distribuições.

Atualmente, vários SDLs no estado-da-arte têm utilizado este tipo de re-segmentação (WOOTERS; HUIJBREGTS, 2008; FREDOUILLE; BOZONNET; EVANS, 2009; LEEUWEN; KONEčNý, 2008; ZHU et al., 2008). Alguns autores chamam este método de Realinhamento de Viterbi, pois ele é capaz de corrigir não apenas erros de segmentação, mas também de agrupamento.

2.7 Estado da Arte

Agora discutiremos alguns dos trabalhos no estado-da-arte nas duas principais tarefas de DL: Segmentação e Agrupamento. Falaremos de técnicas que realizam a segmentação e o agrupamento em passo único e, por último, trataremos dos SDLs criados por laboratórios internacionais que realizam pesquisas na área.

2.7.1 Segmentação de Locutor

Métodos que usam GLR e ΔBIC como medidas de distância entre janelas adjacentes ainda são amplamente utilizados. Em (CHENG; WANG; FU, 2010) os autores utilizam para segmentação o Algoritmo de Janelas Deslizantes com parâmetros configurados para gerar segmentos excessivamente. Entretanto, o agrupamento destes segmentos é compensado utilizando uma técnica de dividir-para-conquistar. Em outra técnica, apresentada em (KOTTI; BENETOS; KOTROPOULOS, 2008), visa-se reduzir o custo computacional dos algoritmos de segmentação por janelas. Para isto, os autores estimaram o tempo médio de um segmento de fala no intuito de mover a janela para a próxima região provável de haver troca de locutor.

Uma extensão ao DistBIC foi apresentada em (KADRI et al., 2006), onde o GLR utilizado na primeira etapa, foi substituído por estatísticas T-Quadrado de Hotteling, em uma abordagem híbrida chamada de DIST^{2}BIC. Em Hachem *et al.* (2010) , os autores publicaram uma forma de calcular a verossimilhança utilizando famílias exponenciais e SVM de 1 classe. Esta forma de cálculo aplicada ao GLR se mostrou robusta, acarretando em melhoras na identificação de pontos de mudança no Algoritmo de Janelas Deslizantes.

2.7.2 Agrupamento

Estimar o número correto de Gaussianas em um modelo de mistura é uma tarefa importante para se evitar problemas de *overfitting* ou *underfitting*. Em (IMSENG; FRIEDLAND, 2009) os autores, ao analisaram o desempenho de uma série de configurações de seu sistema de DL, e encontraram uma alta correlação entre o tamanho dos *clusters* e o número de Gaussianas da mistura que os modelavam. Com isso, criaram um modelo de regressão para predizer o melhor número de Gaussianas de acordo com a quantidade de dados no *cluster*. Em outro trabalho, Han e Narayanan (2008) , propuseram um método AHC utilizando um Modelo de Misturas Gaussianas incremental. Nele, a cada iteração do algoritmo (AHC), a fusão entre os 2 *clusters* mais próximos não acontecia apenas pela união de seus dados, mas também pela "concatenação" dos modelos. Por exemplo, se um *cluster* modelado por 4 Gaussianas fundir-se com outro, modelado por 10 Gaussianas, resultará em um modelo de 14 Gaussianas. Este novo modelo não recebe nenhum tipo de treinamento, apenas reajustes no pesos de todas as Gaussianas.

Outro trabalho importante, com respeito à modelagem de GMMs, é o de Reynolds *et al.* (REYNOLDS; QUATIERI; DUNN, 2000), 2000. Apesar de ser voltado para a Verificação de Locutor, a abordagem de treinamento de GMMs proposta é até hoje amplamente utilizada em DL. O trabalho propõe o treinamento por adaptação de Modelos Universais

(*Universal Background Models* - UBM), que são GMMs de vários componentes previamente treinados com um grande conjunto de dados que inclui muitos locutores. A idéia é que este modelo universal consiga rapidamente (em poucas iterações) ser adaptado aos dados de entrada. A Figura 2.14 ilustra como é realizado o processo de adaptação de um UBM para os dados um locutor. O passo (i) da figura mostra quatro componentes iniciais de um UBM. Ao serem apresentadas aos dados em (ii), as componentes que tem alguma relação com os dados sofrem adaptação para se ajustarem a eles, enquanto as não-relacionadas são eliminadas do novo modelo. A eliminação ocorre de duas maneiras: anulando (igualando a zero) o peso da Gaussiana (isto pode ocorrer durante o treinamento), ou eliminando da mistura as Gaussianas cuja distância de Bhattacharyya para a mesma Gaussiana do UBM (após o treinamento) seja igual a zero.

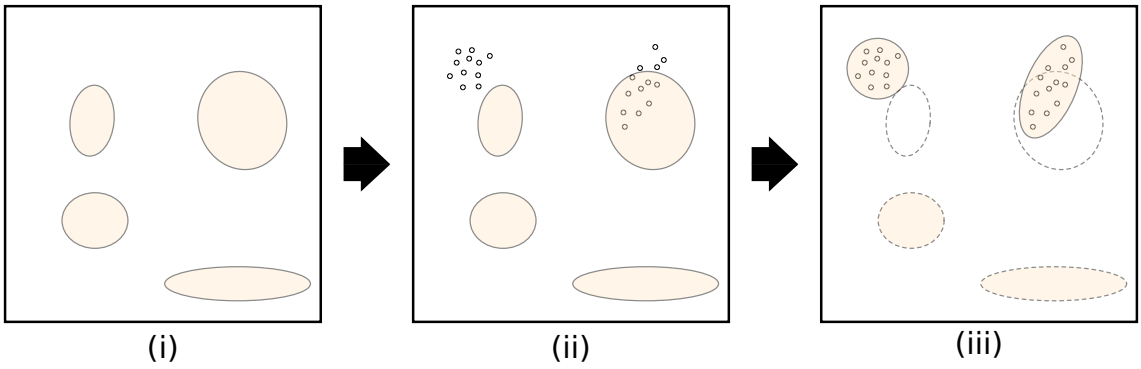


Figura 2.14: Processo de adaptação das componentes de um UBM (ovais na cor laranja) aos novos dados de entrada (círculos pretos pequenos).

2.7.3 Segmentação e Agrupamento em Passo Único

Ajmera e Wooters (AJMERA; WOOTERS, 2003), 2003, propuseram uma abordagem *bottom-up* de segmentação e agrupamento em passo único (em inglês, *one-shot segmentation and clustering*). Nela, a inicialização é feita de maneira uniforme, ou seja, o áudio é quebrado em segmentos de tamanhos iguais, sem uso de algum critério de definição de pontos de mudança. Depois, uma GMM é treinada (ou adaptada) para cada segmento, e um algoritmo de 3 etapas é iniciado: (i) na primeira etapa usa-se o Algoritmo de Viterbi para realinhar os segmentos (permitindo a suavização das fronteiras entre eles, principalmente após a inicialização grotesca); (ii) na segunda, treina-se novamente todas as GMMs de acordo com as novas regiões dos segmentos; (iii) e por último realiza-se a união dos *clusters* mais próximos e, se necessário, retorna-se ao passo (i). Caso na última etapa nenhum *cluster* for unido a outro, então encerra-se o algoritmo. Este método é muito eficaz, e atualmente versões parecidas são utilizadas nos SDLs no estado-da-arte.

Uma técnica *top-down* de segmentação e agrupamento em passo único, chamada de E-HMM (sigla para *Evolutive-HMM*), é utilizada em (FREDOUILLE; BOZONNET; EVANS, 2009). Apesar de inicialmente haver uma segmentação (semelhante ao Dist-BIC), todos os segmentos encontrados são considerados pertencentes ao mesmo *cluster*. A partir daí, o algoritmo realiza divisões e re-segmentações sucessivas. A seguir temos os passos principais desta técnica:

1. Inicia-se com um HMM de apenas 1 estado, modelado por uma GMM treinada com todos os segmentos do *cluster* inicial;

2. Adiciona-se um novo estado ao HMM (representando um novo locutor) retirando-se o maior segmento do *cluster* inicial que tenha ao menos 6 segundos de duração, e treina-se uma GMM para ele;

3. Usando o Algoritmo de Viterbi, realinha-se novamente todos os segmentos. O principal objetivo deste passo é encontrar os segmentos relacionados ao novo locutor;

4. Se a etapa anterior realizou mudanças na constituição dos *clusters*, então todas as GMMs são novamente treinadas, e volta-se ao passo 3;

5. Verifica-se se ainda existem segmentos com mais de 6 segundos no *cluster* inicial. Caso existam, pula-se para o passo 2. Caso contrário, então a condição de parada foi atingida.

2.7.4 Sistemas de Diarização

Nesta seção falaremos brevemente sobre alguns do SDLs no estado-da-arte. Variações da ideia de segmentação e agrupamento em passo único são unanimemente utilizadas por estes sistemas. Todos eles também fazem uso de técnicas de exploração de características multi-canais ou multi-microfones (*beamforming*), que trabalham com informações relativas a localidade da fonte sonora. Entretanto estas técnicas fogem do escopo deste trabalho, e por isso somente trataremos das técnicas relativas ao processamento de um único canal de áudio, ou da soma de todos os demais.

2.7.4.1 AMIDA

O sistema AMIDA (LEEUWEN; KONEčNý, 2008) é dividido em dois estágios. O primeiro é a inicialização, onde são executados, em sequência, um algoritmo VAD (baseado em HMM), um algoritmo de segmentação (Janelas Deslizantes) e um algoritmo de agrupamento (AHC), os dois últimos com ΔBIC. Nesta primeira parte, o parâmetro λ (do ΔBIC) é propositalmente configurado com um valor baixo ($\lambda = 1$). Assim, tanto a segmentação quanto o agrupamento encerram-se com um número maior de segmentos e *clusters* do que o ideal (*over-segmentation* e *over-clustering*), entretanto de maior pureza.

O segundo estágio é um processo iterativo, de segmentação e agrupamento em passo único, inicializado por um conjunto de *clusters* oriundos do AHC do primeiro estágio, sobre as seguintes etapas:

1. Re-Segmentação de Viterbi, onde cada estado é modelado por uma GMM cujo o número de Gaussianas é relativo ao tamanho do seu respectivo *cluster*;

2. Re-treinamento das GMMs dos *clusters* que sofreram modificações;

3. União dos *clusters* mais próximos através do cálculo da distância CLR (*Cross-Likelihood Ratio*);

4. Verificação do valor CLR obtido no passo anterior. Se for menor que um dado limiar, então volta-se ao passo 1. Se não, encerra-se o processo.

Apesar do bom desempenho relatado, o segundo estágio do algoríitmo é computacionalmente custoso. Isto acontece pois, em toda iteração, é necessário treinar novamente as GMMs que sofreram alterações, e recomputar toda a matriz de distâncias.

2.7.4.2 ICSI

O ICSI (WOOTERS; HUIJBREGTS, 2008) é atualmente um dos SDLs mais robustos[4]. Um dos seus principais pontos de destaque é o algoritmo de detecção de fala e não-fala. Normalmente para esta tarefa utiliza-se dois UBMs, um treinado com dados de fala, e outro treinado com dados de não-fala. O sistema do ICSI vai um pouco além, e mescla estes UBMs (treinados com dados externos) com GMMs treinadas com dados do áudio sob avaliação (internos). Desta forma, o algoritmo torna-se mais robusto a variações de canal, de ambiente, ou a dados de fala e não-fala diferentes dos de treinamento dos UBMs.

A DL também é realizada em passo único. Durante a inicialização, o áudio é dividido em k segmentos de tamanho fixo, onde k deve ser muito maior que o número de locutores. O algoritmo principal realiza a segmentação e agrupamento em um passo único, onde GMMs são treinadas para cada *cluster*, e um algoritmo de realinhamento é executado para ajustar as margens dos segmentos de cada *cluster*. Depois, os dois *clusters* mais próximos são selecionados (de acordo com a distância ΔBIC), se a distância for menor do que zero, então eles são juntados e inicia-se uma nova iteração. Caso contrário, encerra-se o algoritmo.

2.7.4.3 LIA

O LIA (FREDOUILLE; EVANS, 2008; FREDOUILLE; BOZONNET; EVANS, 2009) é um SDL resultante da colaboração entre universidades européias. Seu funcionamento é descrito em 3 estágios: Detecção de Fala, segmentação e agrupamento com E-HMM, e re-segmentação. A principal diferença deste sistema para os outros é o uso de uma abordagem *top-down*.

A detecção de fala é feita com um HMM de 2 estados (fala e não-fala), modelados por GMMs de 32 componentes. O principal algoritmo DL do sistema é o E-HMM, descrito anteriormente na Seção 2.7.3. Por fim, uma re-segmentação é realizada com parâmetros normalizados para terem média 0 e variância 1.

2.7.4.4 LIMSI

O LIMSI (ZHU et al., 2008) é um SDL francês que, ao contrário dos sistemas mostrados anteriormente, não realiza a segmentação e o agrupamento em passo único, e sim separados.

A segmentação é obtida por dois algoritmos executados em sequência. Primeiramente realizada-se uma segmentação por Janelas Deslizantes de 5 s. Depois, GMMs diagonais de 8 componentes são treinadas para cada segmento encontrado, e então é feito o Realinhamento de Viterbi com este conjunto de GMMs.

O agrupamento também é feito por duas execuções do AHC. Na primeira usa-se a distância ΔBIC como critério e seleção de parada (todas as distâncias maiores do que zero), onde cada *cluster* sempre é modelado por uma Gaussiana completa. E a segunda execução modela os *clusters* restantes com GMMs de 128 componentes diagonais treinadas por adaptação de UBMs, para então calcular a distância CLR entre todos os pares. O critério de parada é dado por um limiar sobre o CLR.

[4]Possui os erros de diarização mais baixos nas bases tradicionais do NIST.

3 CONFIGURAÇÃO EXPERIMENTAL

Este capítulo descreve a configuração experimental utilizada neste trabalho. A Seção 3.1 descreve as bases de áudio utilizadas. A Seção 3.2 aborda as métricas empregadas para medir o desempenho de SDLs e de seus módulos individualmente. A Seção 3.3 detalha os algoritmos e parâmetros utilizados no SDL proposto, e apresenta os resultados obtidos. Finalmente, na Seção 3.4, apresentamos uma técnica de otimização para o algoritmo AHC, capaz de reduzir significantemente seu tempo de execução, sem afetar o desempenho.

3.1 Bases de Áudio

Em DL, bases de áudio consistem em gravações oriundas de conversações telefônicas, encontros, reuniões, programas de rádio ou televisão, entre outros. Uma base também deve possuir outros aquivos que descrevam o conteúdo das gravações. No caso de *Rich Transcription*, podem haver inúmeros campos de descrição como: marco de início, marco de término, identificação do locutor, sexo do locutor, tipo de áudio (fala, música, ruído, etc.), número de canais, palavras ditas e/ou lexemas, tipo do microfone, entre muitos outros (STANDARDS; , NIST). Na DL apenas os 3 primeiros (marcos de início e término de falas e identificação do falante) são utilizados.

A validação do sistema proposto é realizada utilizando bases de dados de duas fontes: Avaliações do NIST e AMI Corpus.

3.1.1 Avaliações do NIST

Neste trabalho utilizamos 4 bases de áudio do NIST referentes às avaliações de Reconhecimento de Locutor de 2000 (STANDARDS; , NIST) e 2002 (STANDARDS; , NIST) (*Speaker Recognition Evaluation* - SRE). Apesar do nome, umas das tarefas inclusas nestas avaliações era a segmentação e agrupamento de locutores, que posteriormente passou a ser chamada de Diarização de Locutor.

As avaliações de 2000 e 2002 são compostas por quatro bases de áudio, as quais são:

- *NIST Call-Home 2000* (CHOME00): 500 gravações de ligações telefônicas, podendo conter mais do que dois locutores cada uma. A taxa de amostragem do sinal é 8000 Hz;
- *NIST Switchboard 2000* (SWBD00): 1000 gravações de ligações telefônicas com apenas 2 locutores. A taxa de amostragem do sinal é 8000 Hz;
- *NIST Switchboard 2002* (SWBD02): 199 gravações de ligações telefônicas com

apenas 2 locutores, entretanto mais longas do que a SWDB00. A taxa de amostragem do sinal é 8000 Hz;

- *NIST Broadcast News 2002* (BNEWS02): 75 gravações de telejornais, contendo múltiplos locutores. A taxa de amostragem do sinal é 16000 Hz;

A Tabela 3.1 mostra dados estatísticos sobre o tempo de fala em cada gravação, e o tempo total fala das bases considerando ou não sobreposição. Podemos notar que a base SWBD00, apesar de ser a maior em número de testes, não é a maior em tempo total de fala, pois seus áudios são curtos (1 minuto cada). Na versão posterior, SWBD02, os áudios ficaram em média 1 minuto maior. Entretanto o número de testes foi reduzido para 200. As bases SWBD00 e SWBD02 são as únicas cujo tamanho dos áudios (i.e. tempo de fala + não-fala) é fixo, sendo 1 e 2 minutos respectivamente. Já a base CHOME00 possui a maior variação no tempo de fala, podendo durar entre 37 segundos a pouco mais de 9 minutos.

As três primeiras são compostas de gravações de conversas por telefone, e a última de gravações de transmissões de telejornais. Além da qualidade do áudio, uma característica das conversações telefônicas que difere das gravações de telejornais é a alta porcentagem de sobreposição. Em uma conversa típica é comum que um dos locutores não aguarde o outro terminar de falar para iniciar sua vez, já nos telejornais isto é cuidado para que não aconteça. Podemos calcular a porcentagem de sobreposição dividindo a diferença entre o tempo total e o tempo total sem sobreposição, pelo tempo total. Assim, podemos verificar que a base BNEWS02 possui apenas 4,15% de sobreposição, contra 39,73% da base CHOME00, 48,49% da base SWBD00 e 47,53% da base SWBD02.

BASE	**TEMPO DE FALA (s) / TESTE**					**TOTAL (s)**	**TOTAL (s)** (sem sobreposição)
	MÍN.	**MÁX.**	**MÉDIA**	**MEDIANA**	**DESV. PADRÃO**		
CHOME00	36.68	569.42	111.77	70.88	100.76	55884.26	33681.79
SWBD00	31.97	59.90	53.88	54.64	3.82	53884.19	27756.63
SWBD02	34.45	119.99	111.18	113.21	8.98	22125.19	11609.3
BNEWS02	35.52	141.00	115.66	118.55	18.76	8674.51	8314.7

Tabela 3.1: NIST SRE-2000 e 2002: Distribuição dos tempos de fala por teste e total.

A Tabela 3.2 mostra as estatísticas sobre o número de locutores por áudio. Nas bases *Switchboard* o número de locutores é fixado em 2. Entretanto detectamos que um dos áudios da base SWBD02 possui somente 1 locutor. Já as bases CHOME00 e BNEWS02 podem possuir até 7 e 9 locutores, respectivamente. Observando a distribuição destas duas últimas, notamos que, por gravação, a base BNEWS02 possui em média quase o dobro do número de locutores da base CHOME00.

BASE	**LOCUTORES / TESTE**				
	MÍN.	**MÁX.**	**MÉDIA**	**MEDIANA**	**DESV. PADRÃO**
CHOME00	2	7	2.57	2	0.87
SWBD00	2	2	2	2	0
SWBD02	1	2	1.99	2	0.07
BNEWS02	2	9	3.95	4	1.63

Tabela 3.2: NIST SRE-2000 e 2002: Distribuição de número de locutores por teste.

3.1.2 AMI Corpus

O projeto AMI (*Augmented Multi-party Interaction*), criado pelo Instituto de Pesquisas IDIAP[1], visa o desenvolvimento de tecnologias de busca para reuniões. Para isso, foi realizada a construção de uma base de áudio com mais de 100 horas de gravações de reuniões em salas amplamente equipadas para captar sinais de áudio e vídeo. Esta base está disponível para *download*[2] tanto dos arquivos de áudio e vídeo completos, quanto das transcrições.

Dentro do AMI existem 6 bases denominadas EN, ES, IB, IN, IS e TS. As primeiras letras E,I e T referem-se à localização dos ambientes de gravação, sendo "E" para Edimburgo, "I" para IDIAP e "T" para TNO[3]. As segundas letras N, S e B referem-se aos cenários das reuniões, onde N são reuniões naturais (sem a imposição de um tema), S são reuniões com tema imposto (no caso, a discussão sobre a proposta de um novo tipo de controle remoto para TVs) e B são reuniões baseadas em elicitações. Todas estas reuniões possuem sempre 4 participantes.

Neste trabalho utilizaremos a base ES (Edimburgo, com tema imposto), que é a maior delas, contendo 60 gravações de um total de 171, referentes a todas gravações das 6 bases somadas. Além de áudios mais longos, esta base tem quase o dobro do tempo de fala encontrado na nossa maior base do NIST, como pode ser visto na Tabela 3.3.

BASE	TEMPO DE FALA (s) / TESTE					TOTAL (s)	TOTAL (s) (sem sobreposição)
	MÍN.	MÁX.	MÉDIA	MEDIANA	DESV. PADRÃO		
AMI_ES	270.42	2441.84	1534.22	1759.95	526.74	92053.05	66716.73

Tabela 3.3: AMI_ES: Distribuição dos tempos de fala por teste e total.

3.2 Medidas de Desempenho

O objetivo desta seção é descrever as métricas empregadas na avaliação dos experimentos de DL e segmentação de locutor. Para a diarização de locutor, a medida padrão será o Erro de Diarização (*Diarization Error Rate* - DER) utilizada pelas avaliações do NIST. Para a segmentação, a medida utilizada será a Pureza de Cluster, utilizada para avaliar algoritmos de clusterização. As próximas seções descreverão em detalhes estas duas medidas.

3.2.1 Pureza de *Cluster*

Em Gauvain *et al.* (1992) o autor define a Pureza de *Cluster* como sendo a porcentagem (%) de tempo do locutor mais representado pelo *cluster*. Esta medida é utilizada em alguns trabalhos (TRITSCHLER; GOPINATH, 1999; ANGUERA; WOOTERS; HERNANDO, 2006) para quantificar a pureza final após a DL. A Figura 3.1 ilustra, de forma simples, o que acontece com a pureza quando dois segmentos de locutores diferentes são unidos.

Se considerarmos um *cluster* c, contendo fala de N locutores, sendo T_n o tempo de

[1] `https://www.idiap.ch/`

[2] `https://www.idiap.ch/dataset/ami/`

[3] *Dutch Organization for Applied Scientific Research*, (`http://www.tno.nl/`)

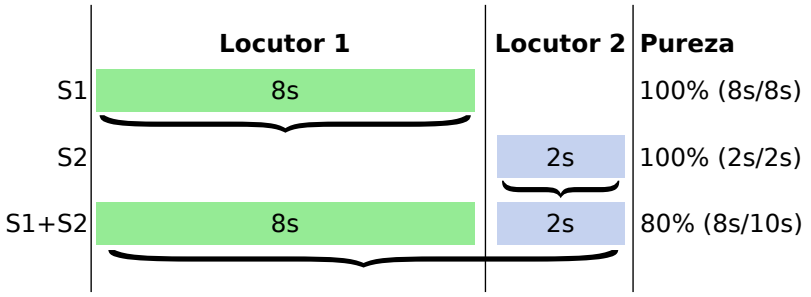


Figura 3.1: Pureza de *Cluster*.

fala de cada locutor, temos então que sua pureza é dada pela formula:

$$p_c = \frac{\max(T_1, ..., T_N)}{\sum_{n=1}^{N} T_n}. \tag{3.1}$$

3.2.2 Erro de Diarização

O DER é a medida utilizada nas avaliações de *Rich Transcription* do NIST. Ele pode ser decomposto pela soma de três medidas diferentes(STANDARDS; , NIST; MIRO, 2006)

- Erro de Locutor (*Speaker Error Time* - SET): tempo de fala que foi erroneamente atribuído a um locutor. Este erro é decorrente de falhas dos algoritmos de segmentação e agrupamento;

- Perda de Locutor (*Missed Speaker Time* - MST): tempo de fala que não foi atribuído a nenhum locutor, ou a menos locutores do que o correto. Esta falha em parte é referente ao módulo de detecção de fala (VAD), que falhou ao considerar regiões de fala como não-fala. Mas também pode ser atribuída aos módulos de segmentação e agrupamento que falharam ao lidar com sobreposição de fala. Por exemplo, se trechos onde há sobreposição de locutores forem atribuídos a menos locutores do que o número total da sobreposição, são considerados MST;

- Falso Alarme de Locutor (*False-alarm Speaker Time* - FST): tempo de não-fala erroneamente atribuído a um locutor. De maneira semelhante ao MST, este erro geralmente decorre de falhas no módulo de detecção de fala, pois este entendeu como fala uma região de não-fala. Também ocorre em trechos equivocadamente considerados como sobreposição, onde julgou-se ter dois ou mais locutores, mas o número correto era menor.

O DER é soma destes erros dividido pelo tempo total:

$$\text{DER} = \frac{\text{SET} + \text{MST} + \text{FST}}{\text{TEMPO TOTAL}}. \tag{3.2}$$

Neste trabalho utilizamos os software *md-eval-v21.pl* para computar o DER. Ele foi escrito na linguagem PERL e é disponibilizado gratuitamente pelo NIST[4].

[4]Disponível para *download* em `http://www.itl.nist.gov/iad/mig/tests/rt/2006-spring/index.html`

3.3 Sistema de Referência

O nosso sistema de referência está dividido em 4 etapas, como mostra a Figura 3.2. Na primeira temos o módulo de parametrização do sinal, que implementa 4 técnicas: MFCC, MEL, LSP e LPCC. Os dados parametrizados são então enviados ao segundo módulo, VAD, que aponta os trechos de fala, com base nas referências. O terceiro módulo, de Segmentação de Locutor, utiliza o algoritmo de Janelas Deslizantes para encontrar pontos de mudança nos dados de fala, retornando os segmentos. E por último, agrupamos estes segmentos de acordo com seu locutor utilizando o método AHC.

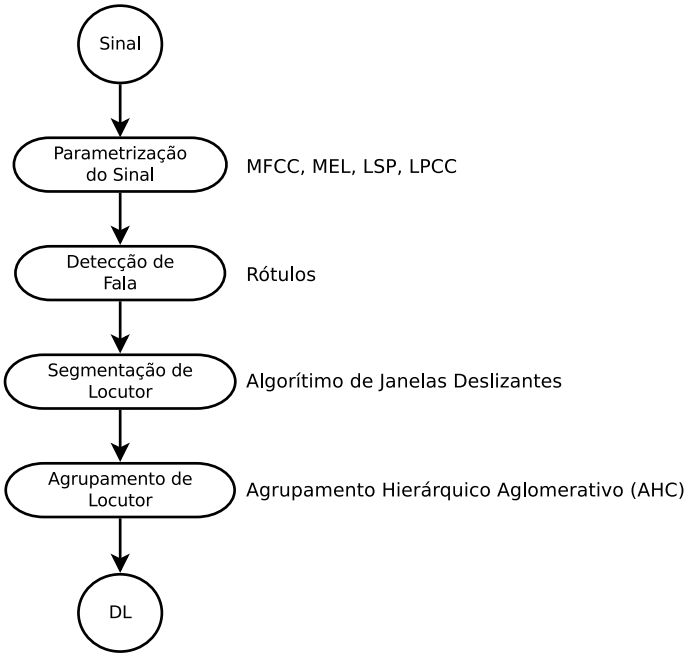


Figura 3.2: Arquitetura do nosso SDL.

A arquitetura escolhida é semelhante à arquitetura padrão, mostrada na Figura 2.1. Entretanto, retiramos o módulo de Re-Segmentação, de forma que a saída do módulo de agrupamento nos dê diretamente o resultado.

Como o objetivo final deste trabalho é comparar diferentes conjuntos de *features* na etapa de agrupamento, nós não usaremos um módulo de VAD, pois este poderia introduzir erros indesejados no sistema e comprometer de alguma forma os resultados. Assim, nosso "VAD" será baseado nos rótulos dos áudios, ou seja, os trechos de fala e não-fala serão recuperadas dos arquivos de referência.

Também descartaremos segmentos com sobreposição de locutor na hora de computar o DER. Isto, somado ao VAD sobre dados rotulados, fará com que o DER seja igual ao SET, pois o MST e o FST serão iguais a zero (0). Assim, sempre que falarmos em DER neste trabalho, iremos na verdade estar nos referindo ao SET.

3.3.1 Segmentação de Locutor

Neste trabalho, o algoritmo de Janelas Deslizantes, descrito na Seção 2.4.2, foi utilizado na etapa de segmentação. Escolhemos este algoritmo pois ele já é utilizado como segmentador inicial em alguns SDLs no estado-da-arte, mostrados na Seção 2.7. E também por necessitar de menos parâmetros de configuração, facilitando a escolha de um bom conjunto de parâmetros.

Para avaliar o algoritmo de Janelas Deslizantes, iremos aplicá-lo na base SWBD02 e medir a pureza final e a quantidade de segmentos encontrados. Para isto utilizamos como parametrização o MFCCs com 19 coeficientes e 30ms de janela, computados a cada 10ms. Esta configuração de parametrização é utilizada por diversos trabalhos na área DL (PARDO; ANGUERA; WOOTERS, 2007; WOOTERS; HUIJBREGTS, 2008; VIJAYASENAN; VALENTE; BOURLARD, 2009; IMSENG; FRIEDLAND, 2009; FRIEDLAND et al., 2009).

Relembrando, o algoritmo de janelas deslizantes possui 4 parâmetros principais:

- Tamanho w (em segundos) das duas janelas adjacentes;
- Deslocamento lateral s (em segundos) das janelas;
- Fator α, que é aplicado sobre o desvio padrão das distâncias entre as janelas (no intuito de se obter um limiar de decisão para máximos locais);
- Medida de distância (e.g. d_{GLR}, d_{KL2}, d_{BAT}).

Impomos a restrição de que o parâmetro s deva sempre ser menor do que w, para que a amostragem não seja pequena em relação ao tamanho da janela. Seguindo esta restrição, e fixando $\alpha = 0.5$, iremos avaliar os segmentos gerados por permutações entre conjuntos de valores para $w \in \{0.5, 1, 2, 4, 8\}$ e $s \in \{0.1, 0.2, 0.3, 0.4, 0.5, 1, 2\}$. Todas as 29 combinações possíveis destes parâmetros são mostradas na Tabela 3.4, onde cada uma recebeu uma identificação (ID).

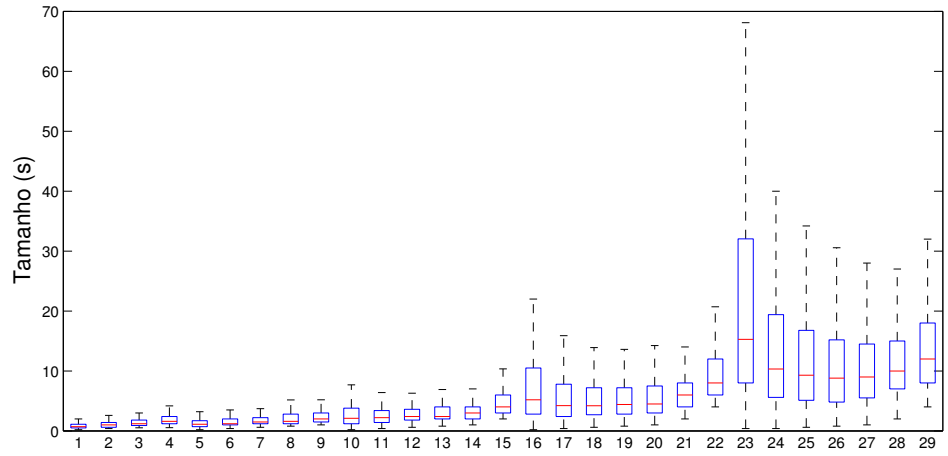
ID	w	s	**ID**	w	s	**ID**	w	s	**ID**	w	s	**ID**	w	s
1	0.5	0.1	**7**	1	0.3	**13**	2	0.4	**19**	4	0.4	**25**	8	0.3
2	0.5	0.2	**8**	1	0.4	**14**	2	0.5	**20**	4	0.5	**26**	8	0.4
3	0.5	0.3	**9**	1	0.5	**15**	2	1	**21**	4	1	**27**	8	0.5
4	0.5	0.4	**10**	2	0.1	**16**	4	0.1	**22**	4	2	**28**	8	1
5	1	0.1	**11**	2	0.2	**17**	4	0.2	**23**	8	0.1	**29**	8	2
6	1	0.2	**12**	2	0.3	**18**	4	0.3	**24**	8	0.2	-	-	-

Tabela 3.4: Parâmetros w e s de cada teste.

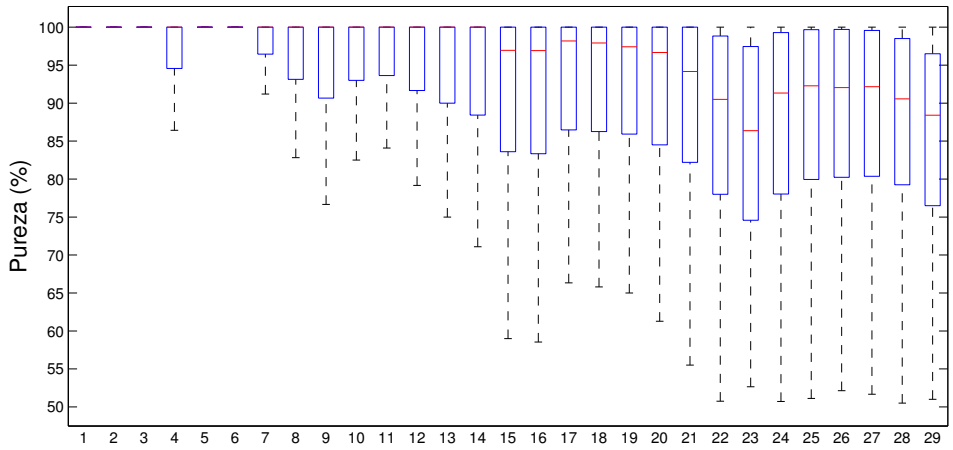
As Figuras 3.3a e 3.3b mostram *boxplots* dos tamanhos e purezas dos segmentos encontrados por todas as 29 configurações do algoritmo na base SWBD02. De imediato podemos notar que a pureza é inversamente proporcional ao tamanho dos segmentos gerados, ou seja, quanto maior um segmento maior é a chance deste conter fala de múltiplos locutores.

Observando também a Figura 3.4, que mostra a frequência [5] com que dois segmentos adjacentes pertencem ao mesmo locutor, podemos inferir com mais clareza a atuação dos parâmetros w e s do algoritmo de segmentação por janelas deslizantes. Primeiramente, uma janela muito pequena implica na geração de um excessivo número de segmentos onde, na menor configuração testada ($w = 0.5s$), podemos encontrar em média até 3 segmentos de um mesmo locutor ocorrendo consecutivamente antes de um ponto de mudança legítimo. Por outro lado, uma janela muito grande ($w = 8s$) tende a conter dados

[5] Número de segmentos adjacentes atribuídos ao mesmo locutor, dividido pelo número total de segmentos (de acordo com a referência).



(a) *Boxplot*: Tamanhos dos segmentos.



(b) *Boxplot*: Pureza dos segmentos.

Figura 3.3: *Boxplots* dos tamanhos e purezas dos segmentos encontrados por todas as 29 configurações do algoritmo de segmentação por Janelas Deslizantes.

de múltiplos locutores, minimizando assim o número de máximos locais significantes que o algoritmo encontra, e diminuindo a possibilidade de se ter segmentos adjacentes de um mesmo locutor.

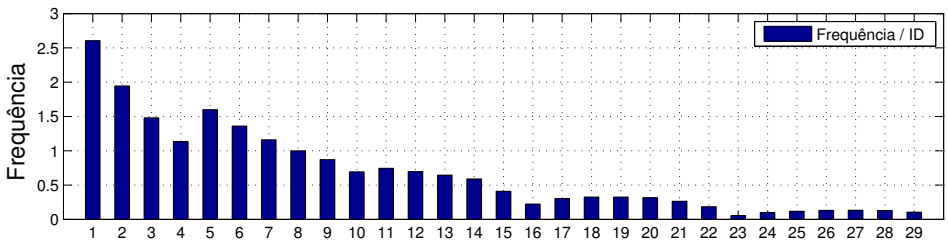


Figura 3.4: Frequência com que segmentos adjacentes pertencem ao mesmo locutor.

No caso do parâmetro de deslocamento s, valores mais baixos aumentam a taxa de

amostragem, acarretando em um maior número de segmentos adjacentes de um mesmo locutor. Isto aumenta a pureza dos segmentos encontrados, ao passo que diminui seu tamanho, pois o algoritmo tende a capturar oscilações excessivamente. Já valores mais altos diminuem a taxa de amostragem, implicando em menos oscilações e podendo acarretar na perda de informações importantes.

A configuração ideal de segmentação é aquela que tenta encontrar um equilíbrio entre o tamanho dos segmentos e a sua pureza. Configurações que favorecem a pureza acabam causando *oversegmentation* (Figura 3.5), onde há uma explosão do número total de segmentos. Isto também gera segmentos muito pequenos contendo informações insuficientes para se obter um modelo estatístico confiável, podendo prejudicar a etapa seguinte de agrupamento. Por outro lado, favorecer o tamanho dos segmentos pode gerar uma grande quantidade de segmentos consideravelmente impuros, e novamente desfavorecendo a criação de modelos estatísticos, desta vez devido ao uso de dados de múltiplos locutores.

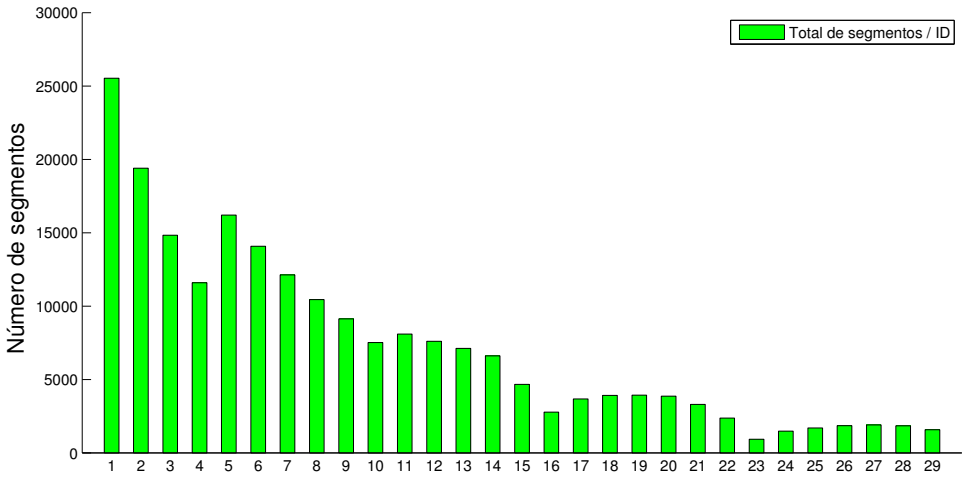


Figura 3.5: Número total de segmentos gerados por configuração.

Em Kotti *et al.* (2008), ao analisar uma base de áudio, os autores inferiram que, durante uma conversação, o tempo médio que uma pessoa fala até dar sua vez para outra é de aproximadamente 3.3s, com desvio padrão de 1.5s. A Figura 3.6 mostra a média e o desvio padrão (em vermelho) do tamanho dos segmentos encontrados por cada configuração de parâmetros de segmentação, e as linhas azuis mostram a média (linha sólida) e o desvio padrão (linhas tracejadas) da distribuição mencionada. Experimentalmente constatamos que uma configuração que gera segmentos nestas proporções tende a obter melhores resultados de diarização. Com base nestes dados optamos então por escolher para nosso sistema de referência a configuração cujos tamanhos dos segmentos mais se aproximassem desta distribuição. Por este critério, selecionamos o conjunto de parâmetros 13 ($w = 2s$ e $s = 0.4s$).

3.3.2 Agrupamento de Locutor

Nesta seção iremos analisar várias configurações do nosso módulo de agrupamento de locutor, que instancia o AHC. Todas as 5 bases de áudio descritas serão utilizadas para as avaliações de desempenho. Apesar dos sistemas no estado-da-arte utilizarem abordagens de segmentação e agrupamento em um passo único, nós optamos por separar estes dois passos, como é geralmente feito nos artigos da área para a demonstração de experimentos.

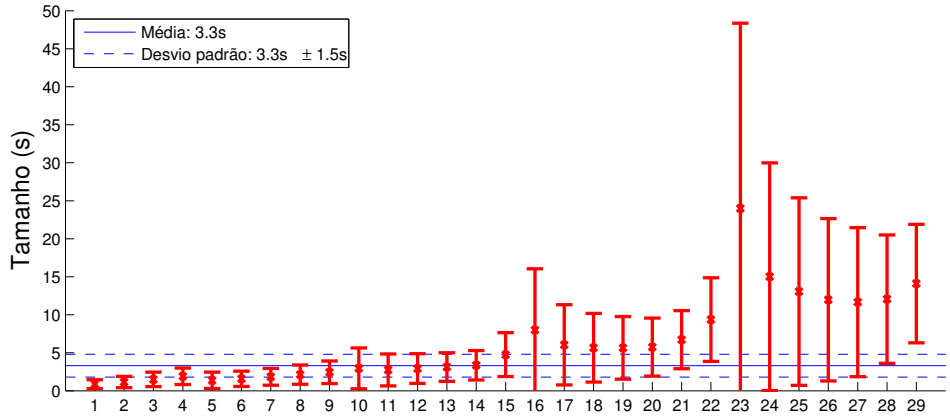


Figura 3.6: Média e desvio padrão do tamanho dos segmentos encontrados por configuração (em vermelho), comparados a análise feita em (KOTTI; BENETOS; KOTROPOULOS, 2008) (em azul).

Com exceção da Seção 3.3.2.4 que analisa o comportamento de critérios de parada, todos os resultados que serão apresentados ao longo deste trabalho consideram que o critério de parada foi otimizado. Sendo assim, o AHC pára sua execução quando o número de *clusters* for igual ao número de locutores do áudio.

Inicialmente, na Seção 3.3.2.1 daremos continuidade a análise da seção anterior, verificando o desempenho do AHC em todas as configurações de segmentação. Na Seção 3.3.2.3 analisaremos o desempenho do AHC com as 4 técnicas de parametrização mostradas no Capítulo 2: MFCC, MEL, LSP e LPCC. Depois, na Seção 3.3.2.4 trataremos de critérios de parada.

3.3.2.1 Análise sob Configurações de Segmentação

Continuando a análise da Seção 3.3.1, mostraremos agora o desempenho obtido pelo método AHC em todos os 29 conjuntos de segmentos. Os dados foram parametrizados com MFCC de 19 coeficientes, calculados sobre janelas de 30 ms espaçadas por 10 ms. A medida de distância utilizadas foi o d_{GLR}.

A Figura 3.7 mostra a o Erro de Diarização (DER) para cada uma das configurações de segmentação da seção anterior (base SWBD02), onde o melhor resultado foi conseguido com a configuração 13 (DER = 15.12%) e o pior com a configuração 23 (DER = 28.19%). Note que as configurações de 1 a 7, notavelmente mais puras que a configuração 13 (ver Figura 3.3b), não conseguiram obter um DER menor. Além disso, ainda obtiveram os piores tempos de execução (Figura 3.8). As configurações de 22 a 29, que possuem janelas maiores (w entre 4s e 8s) e com segmentos consideravelmente impuros, não conseguiram obter bons resultados de agrupamento levando a DERs superiores a 20%, entretanto seus tempos de execução são menores do que a metade do tempo gasto pela melhor configuração[6].

É interessante relembrar que a configuração 13 é também aquela cuja distribuição de tamanhos dos segmentos mais se assemelha aquela fornecida por (KOTTI; BENETOS; KOTROPOULOS, 2008). Por não pertencer diretamente ao escopo deste trabalho,

[6]Neste trabalho todos os experimentos foram executados em uma máquina com processador Intel Core 2 Quad Q8400, 4 Gb de memória RAM e sistema operacional Linux Ubuntu 12.04.

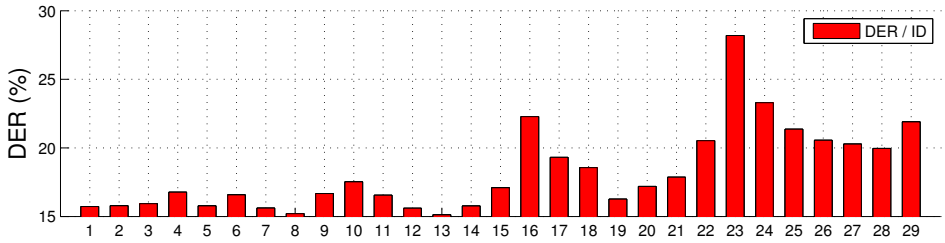


Figura 3.7: DER do agrupamento das 29 configurações de segmentação para a base SWBD02

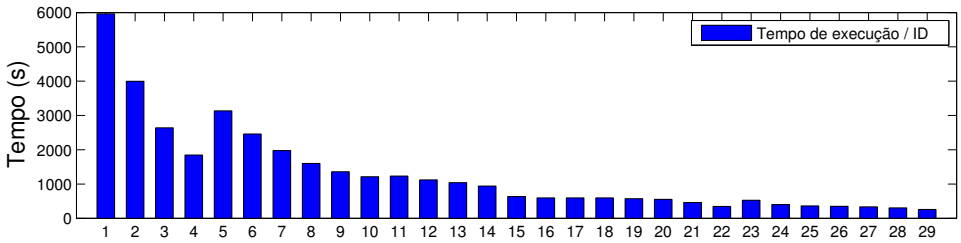


Figura 3.8: Tempo de execução do AHC nas 29 configurações de segmentação para a base SWBD02

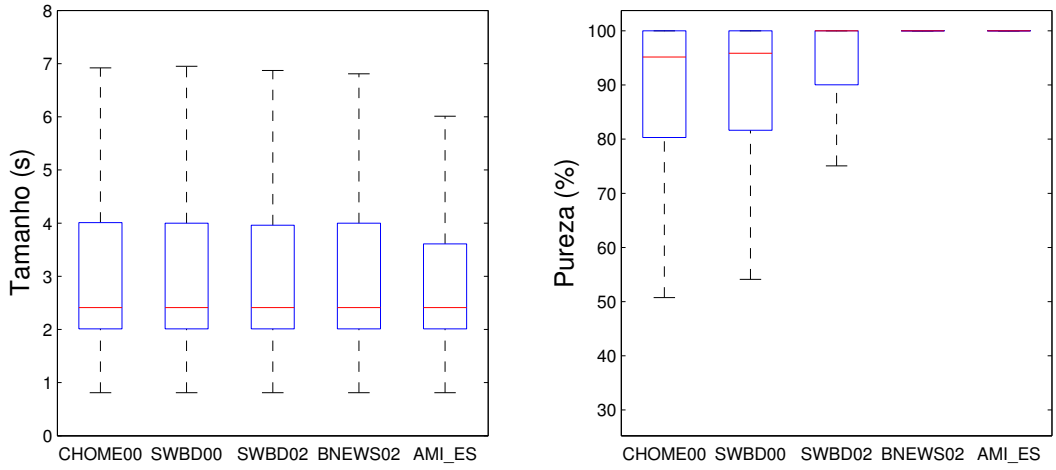
voltaremos a falar sobre este fato durante as conclusões finais.

Tendo visto que a melhor configuração de segmentação para o algoritmo de Janelas Deslizantes, ao menos para a base SWBD02, é a de número 13 ($w = 2s$,$s = 0.4s$,$\alpha = 0.5$,GLR). Adotaremos esta configuração para todo o restante do trabalho. A Figura 3.9 mostra gráficos *boxplot* para o tamanho (Figura 3.9a) e pureza (Figura 3.9b) dos segmentos. Na Figura 3.9b é nítida a vantagem de desempenho do algoritmo de Janelas Deslizantes em áudios de maior resolução (8Khz das bases CHOME00, SWBD00 e SWBD02, contra 16Khz das bases BNEWS02 e AMI_ES) e menos ruídos de canal (bases CHOME00, SWBD00 e SWBD02 são conversações telefônicas). Apesar da Figura 3.9a nos mostrar que, em geral, os segmentos da base AMI_ES são um pouco menores, o que naturalmente elevaria sua pureza, a distribuição dos segmentos da base BNEWS02 é muito próxima das distribuições das outras bases amostradas em 8Khz. Apesar disso, não observou-se queda na pureza dos segmentos desta base.

3.3.2.2 Seleção de Medidas de Distância

Além da inicialização do AHC, dada pela segmentação inicial, outro parâmetro que deve ser definido é a medida de distância utilizada. Neste sentido, executamos o AHC com as distâncias d_{GLR}, d_{GLR_Σ}, d_{KL2}, d_{BAT} e d_{ICR}, onde os resultados são apresentados na Tabela 3.5. A distância d_{GLR} produziu os melhores desempenhos em todas as bases de áudio. O segundo melhor resultado em todas as bases do NIST foi conseguido pela distância d_{ICR}. E os piores resultados, pelas distâncias d_{KL2} e do d_{BAT}.

A principal diferença das distâncias d_{GLR} e d_{ICR} para as distâncias d_{KL2} e d_{BAT}, é que as duas primeiras levam em consideração também a quantidade de dados em cada *cluster*. Esta característica é importante pois, à medida que o AHC é executado, ele passar a unir *clusters*, e começa a ser comum a comparação entre grandes conjuntos de dados contra outros pequenos. É justamente nestes casos em que as medidas d_{KL2} e d_{BAT} sofrem degradação no desempenho. Entretanto é comum ver algoritmos de Segmentação



(a) *Boxplot*:Tamanhos dos segmentos. (b) *Boxplot*: Pureza dos segmentos.

Figura 3.9: *Boxplot* da (a) pureza e (b) tamanho dos segmentos encontrados com o algoritmo de Janelas Deslizantes na configuração ($w = 2s$,$s = 0.4s$,$\alpha = 0.5$,d_{GLR}).

de Locutor utilizando estas métricas (SIEGLER et al., 1997; DELACOURT; KRYZE; WELLEKENS, 2000) pois, por exemplo, quando comparamos duas janelas no algoritmo de Janelas Deslizantes, a quantidade de dados em cada janela é igual.

BASE	CHOME00		SWBD00		BNEWS02		SWBD02		AMI_ES	
	DER	TE (s)	DER	TE (s)	DER	TE (s)	DER	TE (s)	DER	TE (s)
d_{GLR}	**29.26%**	2227.83	**18.16%**	795.02	**14.22%**	228.85	**15.52%**	563.45	**17.73%**	30700.06
d_{GLR_Σ}	**29.26%**	1047.9	**18.16%**	516.9	**14.22%**	120.69	**15.52%**	299.13	**17.73%**	13548.51
d_{KL2}	37.64%	1592.95	32.59%	686.97	42.25%	173.23	31.31%	426.42	59.37%	20763.96
d_{BAT}	37.52%	**723.3**	32.54%	**448.88**	42.22%	**88.83**	31.29%	**230.41**	59.35%	**6552.69**
d_{ICR}	35.69%	4179.46	30.23%	603.56	37.74%	207.16	30.47%	510.71	58.81%	218593.11

Tabela 3.5: DER e Tempo de Execução (TE) do AHC para as medidas de distância d_{GLR}, d_{GLR_Σ}, d_{KL2}, d_{BAT} e d_{ICR}.

Uma constatação negativa sobre o d_{ICR} é o acréscimo de tempo de processamento observado nas bases com áudios longos, como a CHOME00 e a AMI_ES, durante a execução do AHC. No caso da AMI_ES, este acréscimo é superior a 7x o tempo do d_{GLR}. Fazendo uma verificação minuciosa, descobrimos que isso aconteceu devido ao modo como o d_{ICR} lida com grandes e pequenas quantidades de dados. Uma vez que ele mede a quantidade de informação adicionada ao se juntar dois grupos, se um deles for muito maior que o outro, então a junção irá resultar em pouca informação adicionada, mesmo que ambos sejam totalmente diferentes um do outro. Como o AHC funciona selecionando sempre a menor distância, a tendência é que um grupo que se torne grande o suficiente passe a se juntar com os menores em toda iteração. Isto não somente acarreta em um aumento no DER, como também um grande aumento no tempo de execução, pois será necessário a computação de muitas matrizes de covariância contendo enormes quantidades de dados referentes à união de segmentos pequenos com o segmento grande.

Os experimentos com a distância de Bhattacharyya produziram os menores tempos de execução. Isto aconteceu principalmente por que esta, assim como o d_{KL2} e ao contrário do d_{ICR} e do d_{GLR}, não necessita calcular a covariância referente à união dos grupos,

reduzindo em cerca de 1/3 o número de covariâncias computadas. Em segundo lugar no *ranking* de tempo de execução temos o d_{GLR_Σ}. Além de possuir uma execução rápida, ele também obteve os mesmos melhores resultados que o d_{GLR}. Isto é uma evidencia de que a média (GLR_μ, Seção 2.5.2.2) em nada colabora para a separação entre os dois locutores, mas sim a covariância. Neste trabalho então, adotaremos o d_{GLR_Σ} no sistema de referência.

3.3.2.3 Configurações de Parametrização

Nesta seção iremos tratar do uso de outras técnicas de parametrização no AHC. Utilizaremos o MFCC, MEL, LSP e LPCC, todas com a mesma configuração de janelamento (30 ms, espaçadas a cada 10 ms) e número de coeficientes (19).

A Tabela 3.6 mostra o DER resultante do AHC para as 4 técnicas de parametrização em nossa configuração padrão (AHC com d_{GLR_Σ}, e segmentação com algoritmo de Janelas Deslizantes). Apesar do MFCC ser a técnica mais popular, ele obteve os piores resultados em todas as bases, enquanto os melhores foram conseguidos pelas técnicas LPCC e o LSP baseadas em predição linear. No restante deste trabalho, adotaremos o LSP no sistema de referência, pois na média esta foi a técnica que obteve melhores resultados.

BASE	**MFCC**	**MEL**	**LPCC**	**LSP**
CHOME00	29.26%	29.14%	**23.41%**	23.48%
SWBD00	18.16%	17.71%	13.55%	**12.83%**
BNEWS02	14.22%	13.02%	12.51%	**11.71%**
SWBD02	15.52%	15.60%	11.67%	**10.65%**
AMI_ES	17.73%	17.67%	**15.30%**	16.18%
MÉDIA	*18.98%*	*18.63%*	*15.29%*	*14.97%*

Tabela 3.6: Resultados de DL para as técnicas de parametrização MFCC, MEL, LPCC e LSP. Os valores sublinhados indicam que não há diferença estatística para o melhor valor, considerando $\rho = 0.01$.

3.3.2.4 Critério de Parada

O critério de parada mais encontrado em trabalhos de DL é o critério ΔBIC. Como foi visto anteriormente, dois grupos cujo valor $\Delta\text{BIC} < 0$ possivelmente pertencem ao mesmo grupo, caso contrário ($\Delta\text{BIC} \geq 0$) não. O ΔBIC é regulado por um fator λ que, quanto maior seu valor, mais severa é a penalidade aplicada a modelos separados. Em outras palavras, um valor alto de λ favorece a união de grupos no AHC.

Na Figura 3.10 temos o DER de vários valores de λ para as 4 bases do NIST. Nesta experimentação verificamos que os melhores valores para estas bases estão entre 1.3 e 1.5, como mostra a Tabela 3.7. Observando estes valores, podemos ligeiramente notar uma correlação entre o melhor valor para o parâmetro λ e o tamanho médio dos áudios das bases testadas.

Uma outra forma de se obter um critério de parada consiste em estipular um limiar onde, quando todos os valores da Matriz de Dissimilaridade forem maiores do que ele, então o critério de parada foi alcançado. A Figura 3.11 mostra o DER de vários limiares para nossa métrica principal, o d_{GLR_Σ}. Podemos ver que valores entre 1500 e 2500 parecem ser bons valores para as bases do NIST. A Tabela 3.8 mostra os melhores limiares para cada base. Novamente, podemos notar uma ligeira correlação entre o limiar e o

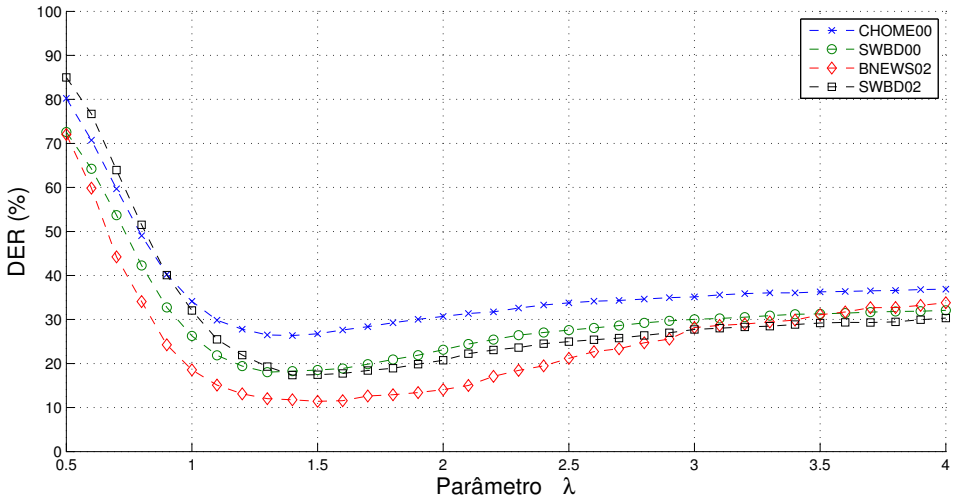


Figura 3.10: DER por base para vários valores λ do critério ΔBIC

BASE	**DER**	λ
CHOME00	26.35%	1.4
SWBD00	18.02%	1.3
BNEWS02	11.4%	1.5
SWBD02	17.38%	1.4

Tabela 3.7: Melhores DERs e seus respectivos valores λ para o critério ΔBIC.

tamanho médio dos áudios das bases.

BASE	**DER**	Limiar GLR_Σ
CHOME00	24.01%	2350
SWBD00	14.79%	1400
BNEWS02	11.02%	2000
SWBD02	11.55%	2250

Tabela 3.8: Melhores DERs e seus respectivos limiares GLR_Σ.

Comparando os resultados das tabelas 3.7 e 3.8, vemos uma melhora substancial quando utilizamos um limiar. A vantagem do limiar também se aplica quando pegamos a média dos DERs de cada base para cada valor do parâmetro λ e do limiar$_{GLR_\Sigma}$. Encontramos que, quando $\lambda = 1.4$, a média dos DERs tem o valor mais baixo igual a 18.43%. E quando limiar$_{GLR_\Sigma} = 1900$, a média dos DERs é 17.43%. Portanto, considerando apenas as bases do NIST, parece ser mais vantajoso o uso do d_{GLR_Σ} com limiar, do que um valor λ para o critério ΔBIC.

A base AMI_ES apresentou um DER diferente das bases do NIST. A Figura 3.12a mostra o DER para vários valores de limiares d_{GLR_Σ}. O melhor DER foi em 14000, o que é um valor muito maior do que os melhores limiares para as bases do NIST (entre 1400 e 2350). Isto se explica pelo fato do valor d_{GLR_Σ} tender a aumentar à medida que se aumenta a quantidade de dados em um *cluster*, conforme análise de Han *et al* (2008) . Uma possível solução para normalizar novamente os limiares, segundo os autores, é a utilização de GMMs.

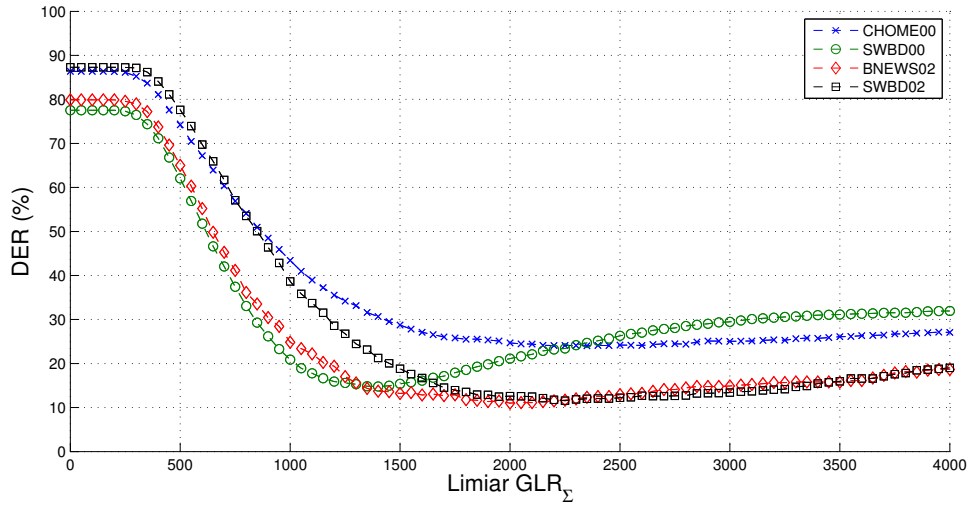
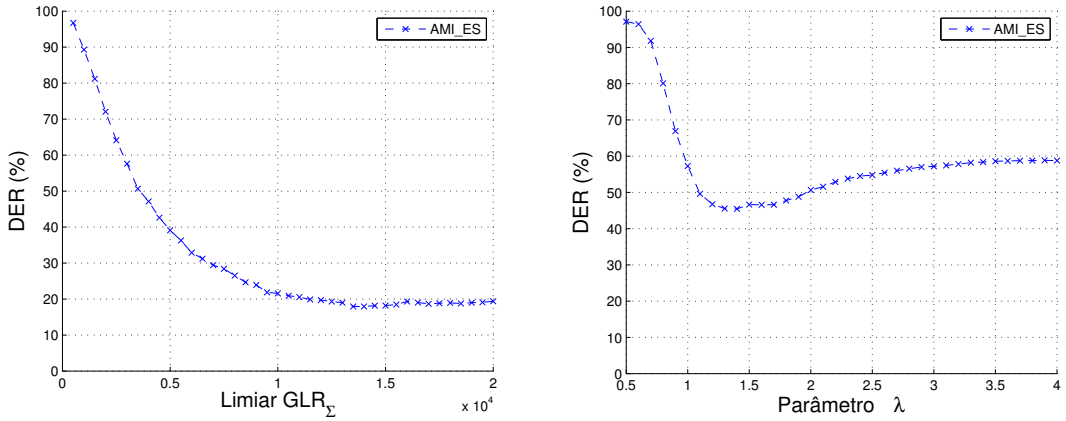


Figura 3.11: DERs para limiares GLR_Σ no intervalo entre 0 e 4000.



(a) Limiares de decisão sobre d_{GLR_Σ}.

(b) Valores λ para ΔBIC.

Figura 3.12: Base AMI_ES: DER para critérios de parada.

Além do limiar, a eficácia do parâmetro λ também é afetada. Na Figura 3.12 o valor λ que obteve o melhor DER foi 1.6. Entretanto, este resultado é altíssimo (45.44%) quando comparado ao DER com número de locutores dados pela referência (16.18%) ou estipulados por um limiar como critério de parada (17.93%). Acreditamos que isso também esteja relacionado ao problema do d_{GLR} tender a aumentar de acordo com o a quantidade de dados. Uma vez que a penalidade BIC cresce logarítmicamente de acordo com o tamanho, o d_{GLR} pode crescer exponencialmente, causando uma disparidade entre ambos.

Neste trabalho, o objetivo maior é a análise do desempenho de técnicas de parametrização após sofrerem transformações baseadas em dados estatísticos, e não a análise da distância ΔBIC ou outro critério de parada. Por esta razão, no restante deste trabalho, iremos considerar como critério de parada um valor otimizado do limiar, de modo que a execução do AHC seja interrompida quando houver um número de *clusters* igual ao número de locutores.

3.4 Otimizações no AHC

Dentre os algoritmos e métodos mencionados e utilizados no nosso SDL, o AHC é aquele que consome maior tempo de execução. Vimos que a complexidade deste método no pior caso é $O(n^2)$, onde, em cada cálculo das distâncias d_{GLR_Σ} ou ΔBIC, é necessário calcular 3 matrizes de covariância (relembrando, uma matriz para cada um dos dois *clusters*, e uma terceira matriz referente a junção deles), terminando em $3n^2$ covariâncias calculadas.

A primeira, e óbvia otimização a se fazer é previamente calcular as matrizes de covariância dos n grupos iniciais do AHC. Desta forma podemos evitar o cálculo das duas primeiras matrizes de covariância referentes aos grupos isolados, e reduzir o número total de matrizes calculadas de $3n^2$ para $n + n^2$. Chamaremos esta otimização de OPT1.

Nesta seção iremos propor uma nova otimização do AHC baseada na decomposição das matrizes de covariância.

3.4.1 Cálculo Eficiente das Matrizes de Covariância

A Matriz de Covariância de um conjunto de dados $\mathbf{X}$ pode ser obtida pela seguinte fórmula:

$$\mathbf{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \tag{3.3}$$

onde $\bar{\mathbf{x}}$ é a média de $\mathbf{X}$.

Pela teoria oriunda da Análise de Discriminantes Lineares (Seção 4.2), se dividirmos $\mathbf{X}$ em dois grupos (ou classes) $\mathbf{X}_i$ e $\mathbf{X}_j$, podemos obter $\mathbf{\Sigma}$ através da soma das matrizes variabilidade intra e inter-classes, $\mathbf{S}_w$ e $\mathbf{S}_b$ respetivamente. Assim:

$$\mathbf{\Sigma} = \mathbf{S}_w + \mathbf{S}_b \tag{3.4}$$

onde:

$$\mathbf{S}_b = \frac{N_i(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})^T + N_j(\bar{\mathbf{x}}_j - \bar{\mathbf{x}})(\bar{\mathbf{x}}_j - \bar{\mathbf{x}})^T}{N_i + N_j} \tag{3.5}$$

$$\mathbf{S}_w = \frac{N_i\mathbf{\Sigma}_i + N_j\mathbf{\Sigma}_j}{N_i + N_j} \tag{3.6}$$

em que $\bar{\mathbf{x}}_i$ e $\bar{\mathbf{x}}_j$ são as médias e N_i e N_j o número de elementos dos conjuntos $\mathbf{X}_i$ e $\mathbf{X}_j$. Como $\mathbf{X}_i \bigcup \mathbf{X}_j = \mathbf{X}$ e $\mathbf{X}_i \bigcap \mathbf{X}_j = \emptyset$, então $N_i + N_j = N$.

Considerando que $\bar{\mathbf{x}} = (N_i\bar{\mathbf{x}}_i + N_j\bar{\mathbf{x}}_j)/(N_i + N_j)$, podemos reconstituir a covariância de $\mathbf{X}$ apenas com os dados das Gaussianas de $\mathbf{X}_i$ e $\mathbf{X}_j$.

Dado que a matriz de covariância da união dos segmentos pode ser estimada através da Equação $\mathbf{\Sigma}_z = \mathbf{S}_b + \mathbf{S}_w$, então a equação 2.24, pode ser reescrita da seguinte forma:

$$d_{GLR_\Sigma}(\mathbf{X}_i, \mathbf{X}_j) = N_z \log|\mathbf{S}_b + \mathbf{S}_w| - N_i \log|\mathbf{\Sigma}_i| - N_j \log|\mathbf{\Sigma}_j| \tag{3.7}$$

Assim, o cálculo da distância d_{GLR} é otimizado, já que a matriz de covariância $\mathbf{\Sigma}_z$ pode ser estimada pelas estatísticas, já calculadas, dos segmentos individuais. Esta nova

otimização reduz o número de matrizes calculadas de $n + n^2$ para n. Entretanto, observamos que o novo $\mathbf{\Sigma}_z$ é muito próximo da covariância de $\mathbf{X}$, mas não exatamente igual. Por este motivo, optamos por calcular a covariância de $\mathbf{X}$ sobre os dados a cada junção do AHC. Desta forma, temos um número final de matrizes calculadas de $2n$. Esta otimização (chamada de OPT2) reduziu em quase 4x o tempo de execução das simulações quando comparado ao OPT1, e em quase 10x sobre o AHC sem nenhuma otimização, como mostra a Tabela 3.9. As pequenas diferenças que aparecem nos DERs são decorrentes das pequenas variações mencionadas, mas não foram significativas (considerando $\rho = 0.05$). Portanto podemos dizer que estatisticamente não há diferença entre os resultados obtidos pelo AHC com e sem nossa otimização.

MÉTRICA	**CHOME00**		**SWBD00**		**BNEWS02**		**SWBD02**		**AMI_ES**	
	DER	**TE (s)**	**DER**	**TE (s)**	**DER**	**TE (s)**	**DER**	**TE (s)**	**DER**	**TE (s)**
GLR_Σ sem otimização	23.48%	1561.55	12.83%	630.7	11.71%	163.97	10.65%	410	16.18%	21654.04
GLR_Σ com OPT1 (ganho)	23.48%	954.42 (1.64x)	12.83%	510.01 (1.24x)	11.71%	114.91 (1.43x)	10.65%	293.01 (1.40x)	16.18%	9012.72 (2.40x)
GLR_Σ com OPT2 (ganho sobre OPT1)	23.33%	**466.8** (2.04x)	12.92%	**369.31** (1.38x)	11.65%	**65.16** (1.76x)	10.66%	**164.61** (1.78x)	16.44%	**2506.41** (3.60x)

Tabela 3.9: DER e Tempo de Execução (TE) para as abordagens de otimização mostradas.

Por conveniência, no restante deste trabalho utilizaremos estas duas otimizações em nosso sistema de referência.

4 REDUÇÃO DE DIMENSIONALIDADE APLICADA À DIARIZAÇÃO

Neste capítulo iremos tratar da aplicação de análises estatísticas para a redução da dimensionalidade em dados de voz. O objetivo é encontrar a menor dimensão possível que preserve a discriminabilidade destes dados. Três técnicas de redução de dimensionalidade serão analisadas: Análise de Componentes Principais (Seção 4.1), Análise de Discriminantes Lineares (Seção 4.2) e Análise de Semi-Discriminantes Lineares (Seção 4.3). Em cada seção, mostraremos os aspectos teóricos da técnica analisada, apresentando exemplos e mostrando seu desempenho quando aplicada na tarefa de DL. Na Seção 4.4, alterações na técnica de Análise de Semi-Discriminantes Lineares são propostas a fim de aumentar o desempenho de diarização do locutor.

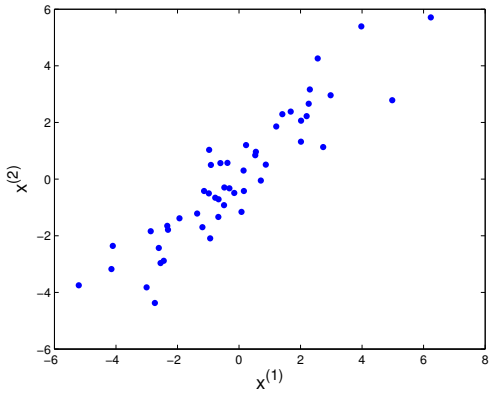
4.1 Análise de Componentes Principais

A Análise de Componentes Principais (*Principal Component Analysis* - PCA) é uma técnica de redução de dimensionalidade não-supervisionada, baseada na dispersão (variância) dos dados. Suas duas principais utilidades são:

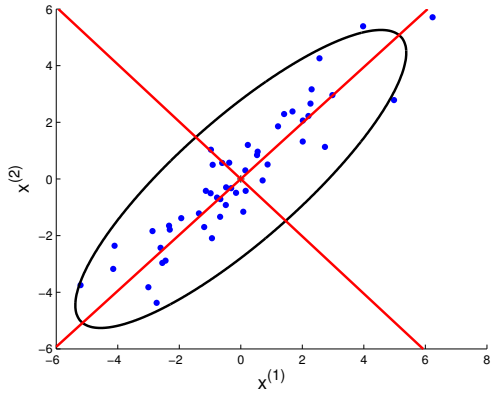
- proporcionar menor tempo de execução e uso de memória em algorítimos de ML;
- permitir a visualização de dados com muitas dimensões em gráficos 2D e/ou 3D.

O PCA projeta os dados em um plano que maximize a variância total. Isto é ilustrado passo-a-passo na Figura 4.1. Em 4.1a, temos um conjunto de dados no qual queremos encontrar os vetores de projeção, e em 4.1b podemos ver a variância destes dados pelo círculo preto (centrado na média), e suas duas componentes principais traçadas em vermelho. Ambas as componentes são ortogonais em relação à outra, e traçadas de maneira a maximizar a variância nos seus respectivos eixos. Finalmente, as Figuras 4.1c e 4.1d mostram como são projetados os dados nas duas componentes principais. Note que a primeira componente principal (Figura 4.1c) melhor representa a variância dos dados, sendo a escolha natural para se reduzir o espaço em $\mathbb{R}^2$ para $\mathbb{R}$ com perdas mínimas. Se aplicarmos todas as componentes principais do PCA, estaremos simplesmente rotacionando os dados, de modo a diagonalizar sua matriz de covariância.

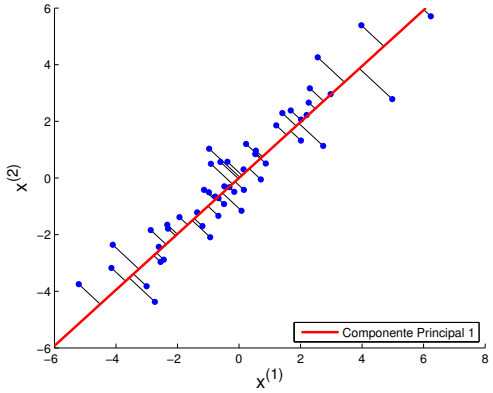
A partir de um conjunto de dados $\mathbf{X}$ pertencente a $\mathbb{R}^n$, queremos reduzir seu número de dimensões de n para k, onde $k \leq n$. Como o PCA é baseado na dispersão dos dados temos, inicialmente, que calcular a matriz $\mathbf{M} \in \mathbb{R}^{n \times n}$ de covariância de $\mathbf{X}$, e encontrar seus autovetores e os correspondentes autovalores. Computacionalmente, isto pode ser



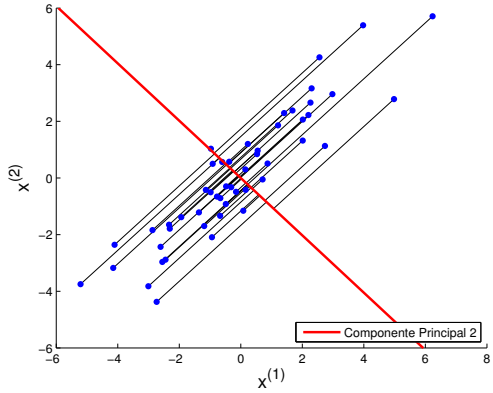
(a) Conjunto de dados em $\mathbb{R}^2$.



(b) Covariância (circulo preto) e componentes principais (linhas em vermelho) dos dados.X



(c) Projeção no 1º componente.



(d) Projeção no 2º componente.

Figura 4.1: Etapas do método PCA.

feito por meio de uma Decomposição em Valores Singulares (*Singular Value Decomposition* - SVD) de $\mathbf{M}$ (HASTIE et al., 2005). Aplicando o SVD podemos decompor $\mathbf{M}$:

$$\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{W}^{\mathbf{T}} \tag{4.1}$$

onde, no caso da matriz $\mathbf{M}$, que é simétrica positiva e semi-definida, $\mathbf{U}$ é uma matriz $n \times n$ cujos vetores coluna são os autovetores de $\mathbf{M}$, $\mathbf{\Sigma}$ é uma matriz diagonal contendo o quadrado dos autovalores de $\mathbf{M}$, e $\mathbf{W}^T = \mathbf{U}$.

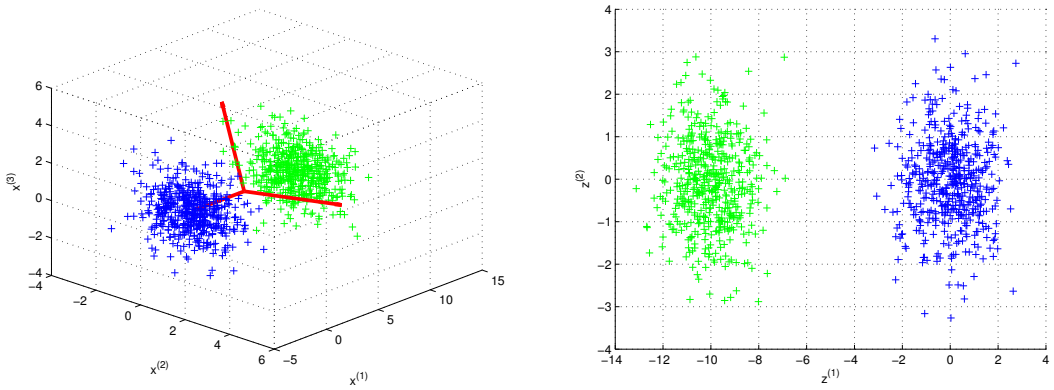
A escolha dos k autovetores que serão utilizados na projeção é feita com base no valor do seu respectivo autovalor. Quanto maior o autovalor, maior é a variância retida pelo autovetor correspondente. Podemos então construir a matriz $\mathbf{T}$ de projeção simplesmente concatenando os k autovetores correspondentes aos k maiores autovalores. O novo conjunto de dados $\mathbf{Z}$, pertencente a $\mathbb{R}^k$, é obtido pela simples multiplicação matricial (DUDA; HART; STORK, 2012):

$$\mathbf{Z} = \mathbf{T}^T\mathbf{X}. \tag{4.2}$$

4.1.1 Exemplos de Visualização

Nesta seção mostraremos 3 exemplos multi-classes onde aplicamos o PCA. Inicialmente utilizaremos dados de 2 e 3 dimensões gerados artificialmente, e depois mostraremos dados de áudio parametrizados de 19 dimensões.

Na Figura 4.2a temos dois conjuntos de dados em $\mathbb{R}^3$ normalmente distribuídos e representados em azul e verde. Ao aplicarmos o PCA, podemos reduzir o espaço de $\mathbb{R}^3$ para $\mathbb{R}^2$ (Figura 4.2b) ainda mantendo a separabilidade entre as classes e facilitando a visualização. Este é um exemplo onde o PCA atua de forma efetiva, reduzindo a dimensionalidade e, consequentemente, a quantidade de memória e cálculos que seriam realizados por algoritmos de ML, e ainda sem acarretar (ou acarretando minimamente) perdas no desempenho.



(a) Dados reais. (b) Projeção final.

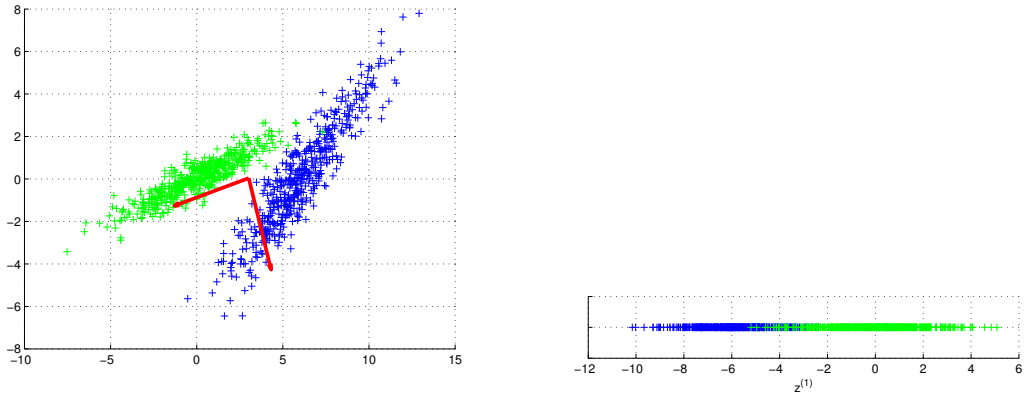
Figura 4.2: Projeção de $\mathbb{R}^3$ (a) para $\mathbb{R}^2$ (b) sobre um conjunto com duas classes de dados (em azul e verde). Em (a), as linhas vemelhas representam os 3 autovetores.

A Figura 4.3 novamente mostra um conjunto com duas classes de dados (em azul e verde), inicialmente em $\mathbb{R}^2$ e posteriormente projetados para $\mathbb{R}$ com PCA. É interessante notar que, na Figura 4.3a há pouca sobreposição entre as classes. Entretanto, a melhor projeção, que maximiza a variância, acarreta o aumento da sobreposição entre as classes. Este é um dos problemas que pode ocorrer com o uso de uma técnica não-supervisionada como o PCA.

Como o escopo deste trabalho trata do processamento de dados acústicos, agora iremos mostrar como exemplo de projeção, um sinal contendo 4 locutores e parametrizado com LSP de 19 coeficientes, extraídos de janelas de 30 ms a cada 10 ms. As figuras 4.4a e 4.4b representam a projeção deste conjunto de dados em $\mathbb{R}^{19}$ para $\mathbb{R}^2$. Na primeira, são mostrados os pontos após a redução, de acordo com seu locutor. Na segunda, os círculos e a cruzes representam o desvio padrão e a média, respectivamente, das distribuições gaussianas dos locutores presentes no áudio. Observando estas figuras fica evidente a dificuldade de se conseguir um bom modelo de locutor de modo não-supervisionado. Vemos que há muita sobreposição entre as classes, tornando praticamente impossível a separação de classes sem o auxilio de informações temporais.

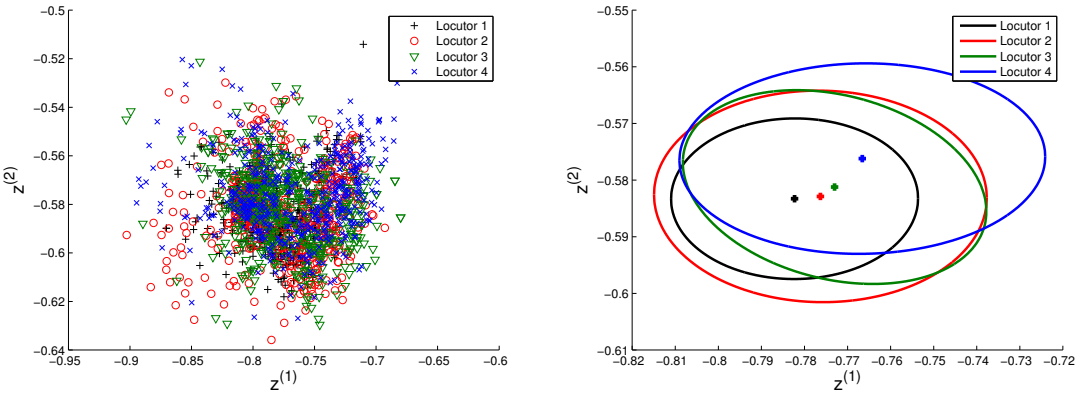
4.1.2 Diarização de Locutor com PCA

Agora aplicaremos o PCA à tarefa de DL. No capítulo anterior utilizamos como técnica de parametrização em nosso sistema de referência, o LSP com 19 coeficientes e



(a) Dados reais. (b) Projeção final.

Figura 4.3: Projecão de $\mathbb{R}^2$ (a) para $\mathbb{R}$ (b) sobre um conjunto com duas classes de dados (em azul e verde). Em (a), as linhas vemelhas representam os 2 autovetores.



(a) Projecão $\mathbb{R}^2$. (b) Desvio padrão da projecão $\mathbb{R}^2$.

Figura 4.4: Projecão com PCA de dados acústicos parametrizados de 19 dimensões.

30ms de janela, extraídos a cada 10ms. Nesta Seção seguiremos utilizando esta mesma configuração, sendo assim, nosso dados possuirão 19 dimensões.

O PCA foi aplicado nos dados da seguinte forma: para cada teste, obtivemos os 19 componentes principais através da matriz de covariância dos dados de fala (os trechos de não-fala foram removidos do cálculo), e então utilizamos o primeiro componente principal para uma dimensão, os dois primeiros componentes principais para duas dimensões, e assim por diante. Os DERs para todas as bases em cada número de dimensões utilizados são mostrados na Tabela 4.1. A tabela também apresenta, ao lado de cada resultado, o nível de significância (ρ) em relação ao resultado obtido pelo sistema de referência (última linha).

Observando a tabela, notamos que o uso de 19 ou menos componentes principais não melhoraram significantemente os resultados de DL (considerando $\rho = 0.05$). Entretanto isto já era esperado dada a natureza do PCA, que preza pela maximização da variância. Em nenhum momento houve melhoras nos resultados com a redução no número de dimensões. Apesar disto, podemos ver que, para todas as 4 bases do NIST, seria possível fixar o número de dimensões em 11 sem que houvesse diferenças significantes nos resultados. Isto implica em uma redução de ao menos 37% ($1 - \frac{12}{19}$) no espaço de memória

DIMENSÕES	CHOME00		SWBD00		BNEWS02		SWBD02		AMI_ES	
	DER	ρ	DER	ρ	DER	ρ	DER	ρ	DER	ρ
1	39.65%	0.00	26.93%	0.00	34.99%	0.00	26.74%	0.00	58.15%	0.00
2	34.94%	0.00	20.91%	0.00	25.44%	0.00	20.27%	0.00	52.12%	0.00
3	32.71%	0.00	19.24%	0.00	23.05%	0.00	17.73%	0.00	47.05%	0.00
4	29.89%	0.00	16.82%	0.00	20.20%	0.00	14.91%	0.00	42.53%	0.00
5	26.32%	0.00	14.99%	0.00	16.60%	0.00	12.85%	0.00	30.52%	0.00
6	25.14%	0.00	14.54%	0.00	14.76%	0.00	12.45%	0.00	26.88%	0.00
7	25.20%	0.00	14.39%	0.00	14.32%	0.00	12.07%	0.00	24.05%	0.00
8	24.00%	0.11	14.06%	0.00	13.85%	0.00	11.09%	0.28	22.01%	0.00
9	23.83%	0.29	13.68%	0.00	14.06%	0.00	11.75%	0.01	20.85%	0.00
10	23.47%	0.98	13.38%	0.05	13.30%	0.00	11.36%	0.08	20.09%	0.00
11	23.70%	0.50	13.16%	0.25	11.97%	0.60	11.08%	0.29	19.99%	0.00
12	23.37%	0.74	12.84%	0.97	12.13%	0.40	11.01%	0.38	18.50%	0.00
13	23.40%	0.81	12.75%	0.78	11.55%	0.75	11.09%	0.28	17.57%	0.00
14	23.38%	0.76	12.79%	0.89	12.18%	0.35	10.65%	1.00	17.77%	0.00
15	23.34%	0.67	12.83%	1.00	**10.76%**	0.05	10.76%	0.79	16.38%	0.32
16	**23.24%**	0.46	**12.71%**	0.67	11.43%	0.57	10.91%	0.52	16.52%	0.09
17	23.54%	0.85	12.83%	1.00	11.52%	0.70	10.96%	0.45	16.76%	0.00
18	23.61%	0.69	12.99%	0.57	11.05%	0.18	**10.32%**	0.41	16.38%	0.32
19	23.48%	1.00	12.83%	1.00	11.71%	1.00	10.65%	1.00	**16.18%**	1.00
SIS. REF.	23.48%	-	12.83%	-	11.71%	-	10.65%	-	16.18%	-

Tabela 4.1: Resultados de DER para reduções de 1 a 19 dimensões com PCA.

ocupado pelos dados, e, consequentemente, melhoras no tempo de execução em algoritmos de Aprendizagem de Máquina. Em nossos testes, o tempo de execução do AHC nestas 4 bases bases chegou a diminuir 32% em média, considerando 11 dimensões e agrupamento com a otimização 1 (OPT1, Seção 3.4.1). Na base AMI_ES este ganho chegou a 10% com 15 dimensões e OPT2. Com OPT1 só há ganho no tempo de execução com 12 ou menos dimensões.

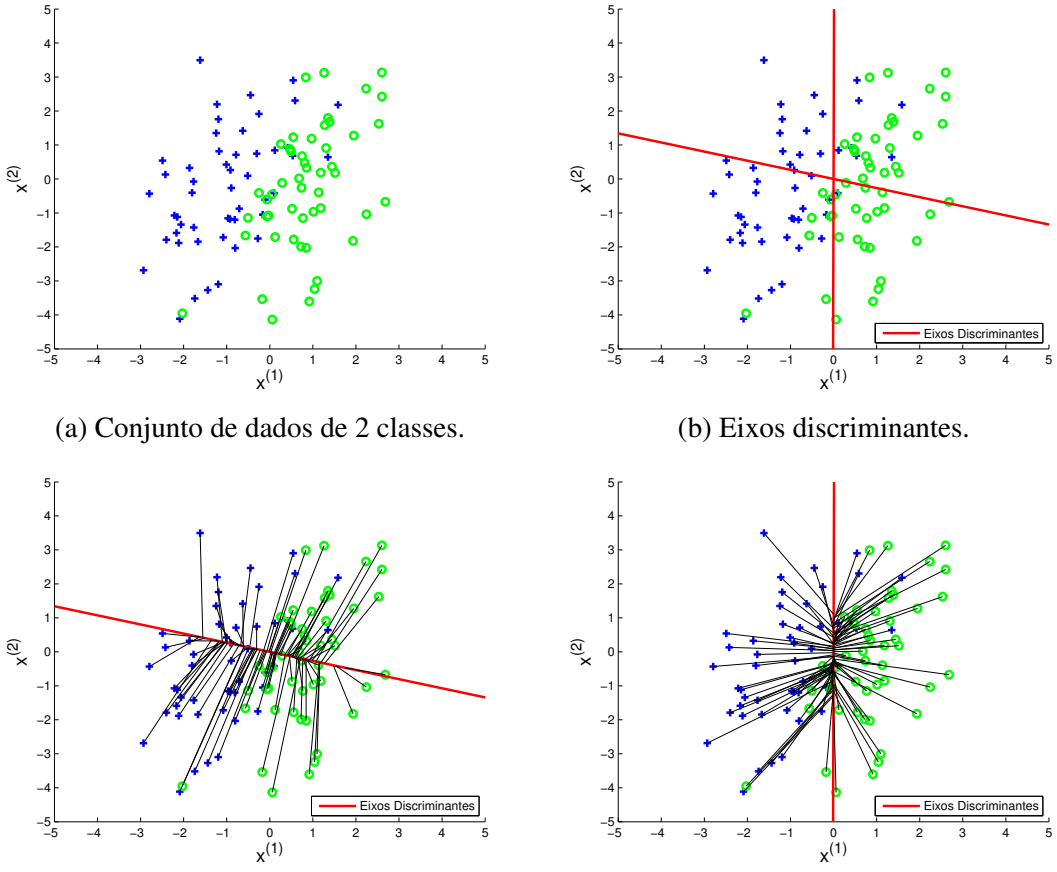
Podemos também observar que os DERs da 19ª dimensão são idênticos aos do sistema de referência. Isto era esperado pois, quando utilizamos todos os componentes principais, estamos apenas rotacionando os dados, de modo a diagonalizar a covariância.

4.2 Análise de Discriminantes Lineares

A Análise de Discriminantes Lineares , em inglês *Linear Discriminant Analysis* (LDA) ou *Fisher Linear Discriminant Analysis* (FLD), é um método de classificação que baseia-se no centroide e na variância das classes. Seu objetivo é encontrar o melhor hiperplano de separação que minimize a sobreposição entre elas.

O LDA também pode ser usado para redução de dimensionalidade se projetarmos os dados em um plano ortogonal ao hiperplano de separação, onde os novos eixos são chamados de *eixos discriminantes* ou *variáveis canônicas*. A Figura 4.5 ilustra os passos do método LDA para projeção. Inicialmente, é necessário que os dados estejam rotulados por classes, como na Figura 4.5a, pois este é um método supervisionado. Sabemos que o hiperplano de separação entre as classes possui o mesmo número de dimensões que os dados. Da mesma forma, o plano ortogonal também possuirá. Isto significa que podemos encontrar um número de eixos discriminantes igual ao número de dimensões dos dados. No exemplo da figura temos duas classes de duas dimensões, portanto o método nos retornará 2 eixos discriminantes (Figura 4.5b). É fácil notar que o primeiro eixo (Figura 4.5c) melhor representa a separação dos dados, e não sua dispersão (como é o caso do PCA). Já o segundo eixo discriminante (Figura 4.5d), apesar do nome, nada nos diz sobre a separação.

Para um conjunto de dados $\mathbf{X}$ separado em L classes, existem três matrizes de dis-



(a) Conjunto de dados de 2 classes.

(b) Eixos discriminantes.

(c) Projeção no eixo com maior autovalor.

(d) Projeção no eixo com menor autovalor.

Figura 4.5: Etapas do método LDA.

persão necessárias para a formulação do critério de separação no LDA (FUKUNAGA, 1990):

- Matriz da variabilidade inter-classes, que mede a dispersão entre os centróides das classes, dada por:

$$\mathbf{S}_b = \sum_{l \in L} P_l (\bar{\mathbf{x}}_l - \bar{\mathbf{x}})(\bar{\mathbf{x}}_l - \bar{\mathbf{x}})^T \tag{4.3}$$

  onde P_l é a probabilidade de uma amostra pertencer à classe l, $\bar{\mathbf{x}}_l$ é a média dos dados na classe l, e $\bar{\mathbf{x}}$ a média geral;

- Matriz da variabilidade intra-classes, que mede a dispersão dos dados em relação ao valor esperado de suas respectivas classes:

$$\mathbf{S}_w = \sum_{l \in L} P_l E[(\mathbf{x}_l - \bar{\mathbf{x}}_l)(\mathbf{x}_l - \bar{\mathbf{x}}_l)^T]. \tag{4.4}$$

- Matriz de covariância de $\mathbf{x}$:

$$\mathbf{\Sigma} = E[(\mathbf{x} - \bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})^T] \tag{4.5}$$

  que mede a dispersão total dos dados sem considerar nenhuma classe. É interessante notar que a soma das matrizes de variabilidade inter e intra-classes resultam na matriz de covariância, isto é, $\mathbf{\Sigma} = \mathbf{S}_w + \mathbf{S}_b$.

O critério de separação deve ser formulado de maneira a maximizar a separação inter-classes ($\mathbf{S}_b$) e minimizar a separação intra-classes ($\mathbf{S}_w$). Assim, para encontrar os eixos discriminantes, devemos procurar pela base vetorial $\mathbf{W}$ que obedeça ao critério (HASTIE et al., 2005):

$$\max_{\mathbf{w}} \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}. \tag{4.6}$$

A tarefa de encontrar $\mathbf{W}$ pode ser resolvida como um problema de autovetores e autovalores:

$$(\mathbf{S}_w^{-1} \mathbf{S}_b) \mathbf{\Phi} = \mathbf{\Lambda} \mathbf{\Phi} \tag{4.7}$$

onde $\mathbf{\Phi}$ é a matriz de autovetores e $\mathbf{\Lambda}$ a matriz de autovalores de $\mathbf{S}_w^{-1} \mathbf{S}_b$. Os eixos discriminantes são dados pelos vetores coluna de $\mathbf{\Phi}$ ranqueados de acordo com seu respectivo autovalor.

A projeção dos dados nos eixos discriminantes é feita da mesma forma que no PCA. Devemos construir a matriz $\mathbf{T}$ de transformação concatenando os k autovetores com maior autovalor, e multiplicar sua transposta por $\mathbf{X}$, assim $\mathbf{Z} = \mathbf{T}^T \mathbf{X}$. Em (FUKUNAGA, 1990) o autor mostra que a projeção em $L - 1$ eixos discriminantes é suficiente para que não haja nenhuma perda de informação de classificação. Isto acontece por que as L funções de densidade de probabilidade são linearmente independentes.

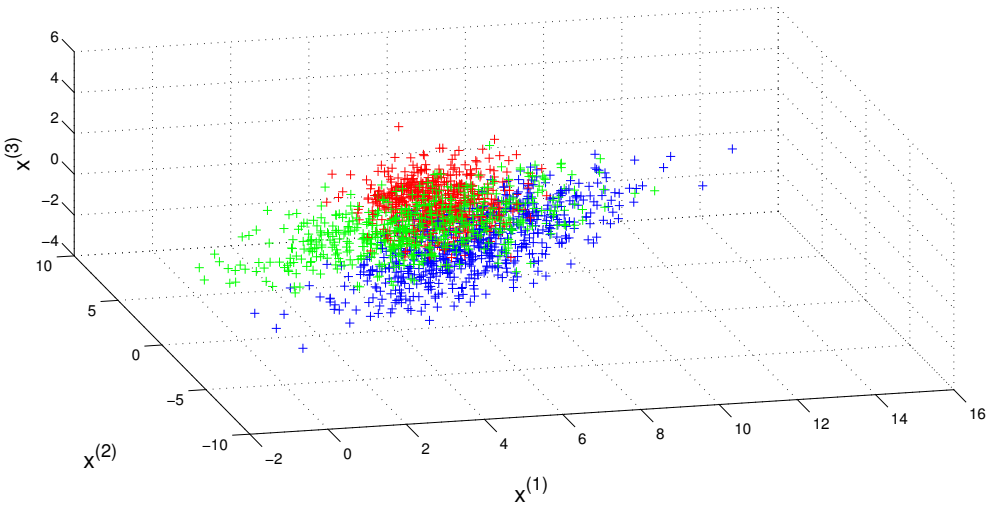
4.2.1 Exemplos de Visualização

Na Figura 4.6a temos um conjuntos de dados tridimensionais com 3 classes (representadas em vermelho, verde e azul), onde todas as classes estão relativamente próximas umas das outras, havendo alguma sobreposição. Ao aplicarmos o PCA, Figura 4.6c, e pegarmos suas duas componentes principais, podemos notar que a maximização da variância obtida por elas não ajuda no processo de discriminação de classes. Neste caso, as classes azul e verde estão praticamente misturadas. Quando executamos o algoritmo EM para treinar uma mistura com 3 Gaussianas, podemos ver que ele ainda consegue capturar corretamente as 3 classes. Entretanto a sobreposição das classes verde e vermelha com a classe azul é grande, conflitando com o desvio padrão.

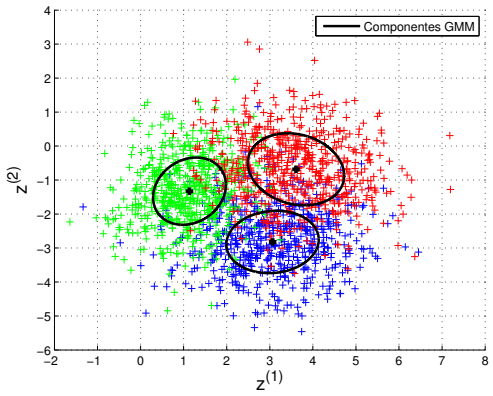
Em casos como este é muito conveniente a aplicação do LDA, Figura 4.6b, onde observamos que as mesmas 3 classes ficaram nitidamente separadas com a projeção nos 2 principais eixos discriminantes. E, ao contrário dos dados projetados com PCA, podemos ver nesta figura que o algoritmo EM conseguiu, com sucesso, separar todas as 3 classes sem nenhuma sobreposição entre as gaussianas.

Na Seção 4.1.1, mostramos também um exemplo baseado em dados de áudio parametrizados com LSP, que estão na Figura 4.4. Agora mostraremos como estes mesmos dados se comportam ao utilizarmos o LDA. Primeiramente, sabemos que os dados possuem 4 locutores, portanto, 4 classes. Assim estes dados, teoricamente, podem ser representados em 3 dimensões (ou seja, projetados nos 3 principais eixos discriminantes) sem nenhuma perda de informação de classificação. Para visualizar melhor estes 3 eixos optamos por mostra-los em duas imagens bi-dimensionais, sendo a primeira para os eixos 1 e 2, e a segunda para os eixos 3 e 4, conforme é mostrado na Figura 4.7.

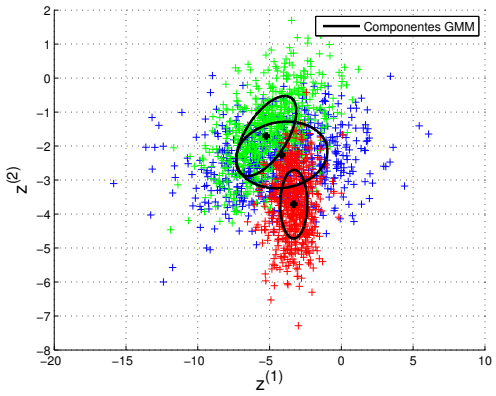
Observando os pontos da Figura 4.7a é possível perceber alguma distinção entre as classes - apesar da grande quantidade de sobreposição - o que não acontece na Figura 4.4a, onde utilizamos o PCA. Na Figura 4.7b, com exceção da classe referente ao locutor



(a) Dados em 3 dimensões.



(b) LDA: 2 principais eixos discriminantes.



(c) PCA: 2 componentes principais.

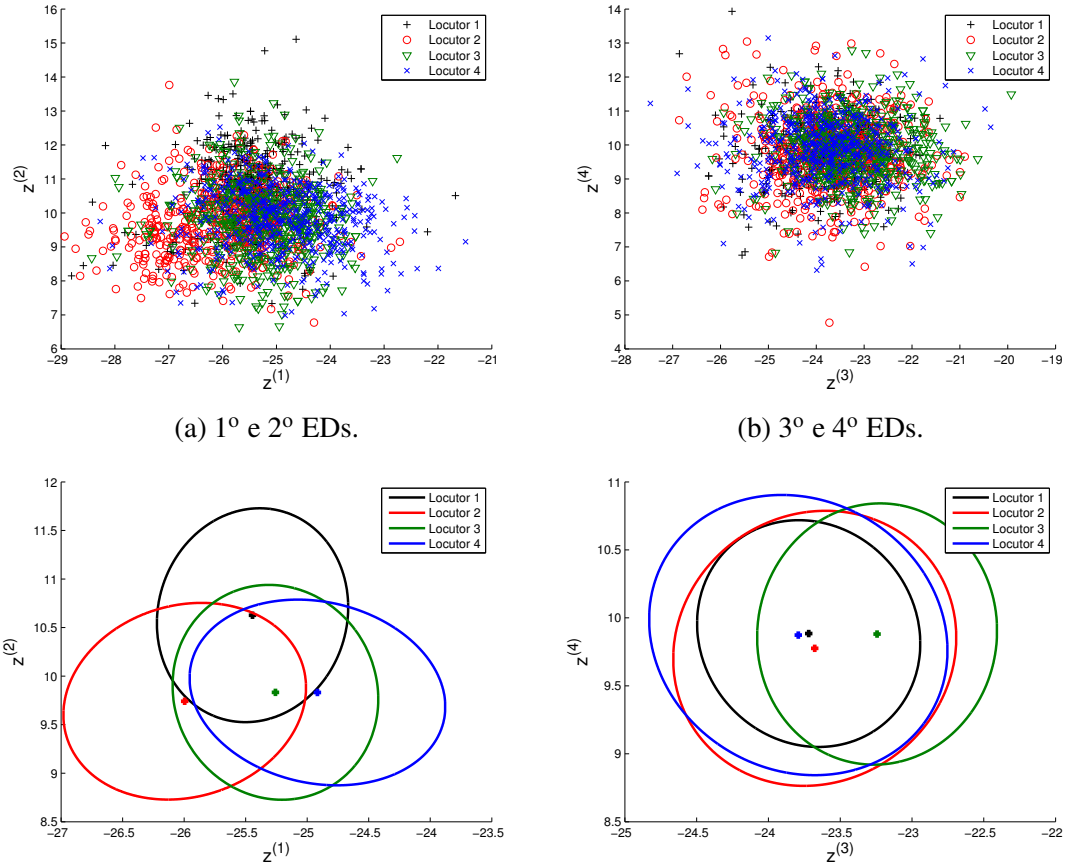
Figura 4.6: LDA.

3, há muita sobreposição entre as classes e fica impossível distingui-las. Assim, para um melhor entendimento e visualização por parte do leitor, mostramos as distribuições Gaussianas de cada classe nas figuras 4.7c e 4.7d. Olhando atentamente estas Gaussianas, podemos perceber que os 3 primeiros eixos ($z^{(1)}$,$z^{(2)}$ e $z^{(3)}$) fornecem informações mais discriminantes quando comparadas aos demais eixos, para 2 locutores. Por exemplo: o eixo $z^{(1)}$ destaca-se na separação dos locutores 2 e 4, o eixo $z^{(2)}$ separa o locutor 1 dos demais, e o eixo $z^{(3)}$ separa o locutor 3 dos demais. Como era esperado, o eixo $z^{(4)}$ em nada colabora com a separação de classes.

4.2.2 Diarização de Locutor com LDA

O LDA não pode ser aplicado na tarefa de DL, pois uma das premissas da tarefa é que não há informação *a priori* sobre os locutores. Assim, a informação dos locutores foi utilizada para a aplicação do LDA a fim de avaliarmos o seu uso nesta tarefa.

Os dados das classes (representando cada locutor) foram selecionados a partir das transcrições das conversas e utilizados para estimar as matrizes de variabilidade intra e inter-classes. Então, os autovetores (ou eixos discriminantes) resultantes da equação 4.7, ordenados por seus respectivos autovalores, foram utilizados para projetar os dados.



(a) 1º e 2º EDs.

(b) 3º e 4º EDs.

(c) Desv. Padrão das classes no 1º e 2º EDs.

(d) Desv. Padrão das classes no 3º e 4º EDs.

Figura 4.7: Projeção dos 4 primeiros eixos discriminantes (ED) de vetores características de 19 dimensões.

Assim como na Seção 4.1.2, para uma dimensão estaremos utilizando apenas o primeiro autovetor, para duas dimensões o primeiro e segundo autovetores, e assim por diante.

Os resultados da diarização utilizando os dados projetados com LDA são apresentados na Tabela 4.2. Os resultados mostram que a utilização do LDA fornece um aumento de desempenho em todas as bases, mesmo considerando um $\alpha = 0.01$ (na tabela, ρ_1 é o valor-p sobre o sistema de referência). Na base BNEWS02 a melhora relativa para 4 dimensões foi de 27.5%. Com relação ao espaço de memória ocupado pelos dados, a redução foi de 74% (CHOME00), 95% (SWBD00), 79% (BNEW02), 95% (SWBD02) e 84% (AMI_ES) nas dimensões de menor DER. A menor dimensionalidade também trouxe melhores tempos de execução nestas dimensões, chegando a reduzir em 38% (CHOME00), 24% (SWBD00), 37% (BNEW02), 40% (SWBD02) e 58% (AMI_ES).

O sistema obteve melhores resultados nas bases SWBD00, SWBD02 e AMI_ES justamente nas dimensões onde teoricamente seriam as ideais, ou seja, as dimensões correspondentes ao número de locutores subtraído de 1. Note que estas 3 bases são as únicas cujo número de locutores por gravação é fixo (2,2 e 4, respectivamente). Nas bases CHOME00 e BNEWS02, onde o número de locutores pode variar, observou-se que, nos melhores resultados, o número de dimensões é próximo da média do número de locutores. Dado o valor-p do DER do melhor resultado para com todos os outros (ρ_2 na tabela), e considerando um $\alpha = 0.05$, podemos ainda reduzir em uma unidade a dimensionalidade destas duas últimas bases sem que haja diferença significante nos resultados.

DIM.	CHOME00			SWBD00			BNEWS02			SWBD02			AMI_ES		
	DER	ρ_1	ρ_2	DER	ρ_1	ρ_2	DER	ρ_1	ρ_2	DER	ρ_1	ρ_2	DER	ρ_1	ρ_2
1	29.65%	.00	.00	**10.91%**	.00	-	22.16%	.00	.00	**9.36%**	.00	-	37.10%	.00	.00
2	23.73%	.44	.00	11.08%	.00	.52	11.91%	.69	.00	10.01%	.11	.09	24.14%	.00	.00
3	21.94%	.00	.83	11.07%	.00	.55	8.68%	.00	.66	9.73%	.02	.34	**15.58%**	.00	-
4	22.00%	.00	.68	11.59%	.00	.01	**8.49%**	.00	-	9.99%	.10	.10	15.70%	.02	.55
5	**21.87%**	.00	-	11.58%	.00	.01	9.36%	.00	.05	9.70%	.02	.38	16.37%	.35	.00
6	22.60%	.01	.02	11.66%	.00	.01	9.26%	.00	.08	10.16%	.22	.04	16.51%	.10	.00
7	22.01%	.00	.66	11.76%	.00	.00	9.31%	.00	.06	10.01%	.11	.09	16.81%	.00	.00
8	23.08%	.22	.00	11.96%	.00	.00	9.19%	.00	.11	9.80%	.03	.25	16.72%	.01	.00
9	22.13%	.00	.42	11.93%	.00	.00	9.53%	.00	.02	10.21%	.27	.03	16.79%	.00	.00
10	22.90%	.07	.00	12.42%	.15	.00	10.49%	.01	.00	10.55%	.80	.00	16.60%	.04	.00
11	22.68%	.01	.01	12.47%	.20	.00	10.72%	.04	.00	10.68%	.94	.00	16.98%	.00	.00
12	23.07%	.21	.00	12.73%	.72	.00	10.34%	.00	.00	10.62%	.94	.00	16.31%	.52	.00
13	23.31%	.60	.00	12.48%	.21	.00	10.91%	.10	.00	10.34%	.44	.01	16.00%	.37	.04
14	22.04%	.00	.59	12.82%	.97	.00	11.42%	.56	.00	10.28%	.36	.02	17.38%	.00	.00
15	22.68%	.01	.01	12.76%	.81	.00	11.23%	.33	.00	9.97%	.09	.12	16.22%	.84	.00
16	23.16%	.33	.00	12.83%	1.00	.00	10.60%	.02	.00	10.55%	.80	.00	15.84%	.09	.19
17	23.31%	.60	.00	12.76%	.81	.00	10.92%	.11	.00	10.65%	1.00	.00	17.89%	.00	.00
18	23.63%	.65	.00	12.93%	.73	.00	11.11%	.22	.00	10.29%	.37	.02	16.59%	.04	.00
19	23.48%	1	.00	12.83%	1	.00	11.71%	1	.00	10.65%	1	.00	16.18%	1	.00
S.R.	23.48%	-	.00	12.83%	-	.00	11.71%	-	.00	10.65%	-	.00	16.18%	-	.00

Tabela 4.2: DER para reduções de 1 a 19 dimensões, com LDA. Onde ρ_1 é o valor-p em relação a referência, e ρ_2 é o valor-p em relação ao melhor resultado.

Na última dimensão (19ª), onde todos os autovetores são utilizados, os resultados são idênticos ao sistema de referência. Assim como no PCA, isto acontece pois, neste caso, houve apenas a simples rotação dos dados.

Na Tabela 4.3 temos o DER para aplicações do LDA em diferentes parametrizações, onde, em cada gravação, os dados foram transformados para sua dimensionalidade ideal ($L-1$ dimensões). Assim, os valores referentes ao LSP nas bases SWBD00, SWBD02 e AMI_ES são os mesmos da tabela anterior (Tabela 4.2) para 1 e 3 dimensões, pois nestas bases o número de locutores por teste é fixo em 2, 2 e 4 respectivamente.

Novamente o LSP obteve os melhores resultados em todas as bases, exceto na CHOME00, a qual também não foi significativamente pior do que o MEL. Surpreendentemente, neste teste o MFCC obteve resultados parecidos com o LSP, sendo significativamente pior apenas na base SWBD00.

PARAM.	DER				
	CHOME00	SWBD00	BNEWS02	SWBD02	AMI_ES
LPCC	22.39%	**10.91%**	**9.01%**	9.95%	16.23%
LSP	22.25%	**10.91%**	**9.01%**	**9.36%**	**15.58%**
MEL	**21.99%**	12.01%	9.74%	10.59%	18.01%
MFCC	22.39%	11.75%	9.64%	9.68%	15.66%

Tabela 4.3: DER para reduções com LDA em $L-1$ dimensões. Resultados sublinhados não são significativamente piores que os melhores (dados em negrito), considerando $\rho = 0.05$.

4.3 Análise de Semi-Discriminantes Lineares

A Análise de Semi-Discriminantes Lineares (*Fisher Linear Semi-Discriminant Analysis* - FLsD), proposta por Giannakopoulos e Petridis (2012) , é um método baseado

em LDA aplicado a DL. Este método aproveita informações temporais dos dados para tentar estimar as matrizes de variabilidade inter e intra-classe, de modo totalmente não-supervisionado. Nesta seção mostraremos como o FLsD é definido, explicaremos o pré-processamento dos dados por janelas de textura, e depois analisaremos os resultados de sua aplicação em nosso sistema de referência.

4.3.1 Definição

Quando analisamos um sinal de áudio referente a uma conversa entre pessoas, podemos pressupor que, para uma dada amostra, todos os seus vizinhos dentro uma janela relativamente pequena, irão pertencer ao mesmo locutor. O FLsD assume esta suposição como verdadeira para formular o conceito de *class-threads* ou, como chamaremos neste trabalho, sub-classes. Cada sub-classe contém um pequeno trecho de amostras temporalmente adjacentes que, teoricamente, pertencem a apenas um locutor. Logo, para cada classe c referente a um dos locutores do áudio, e cada sub-classe v, podemos determinar uma função sobrejetora $h(v) \rightarrow c$, que mapeia toda sub-classe em sua respectiva classe.

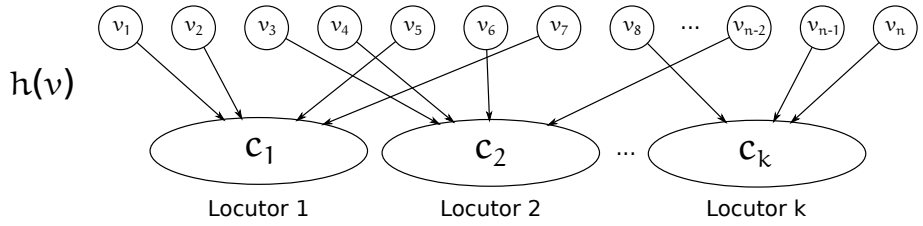


Figura 4.8: Mapeamento de n sub-classes em k classes ou locutores.

A obtenção das sub-classes é um processo simples, semelhante a uma segmentação uniforme, onde o áudio é dividido em pequenas partes de tamanho fixo. Cada uma destas partes é considerada uma sub-classe. O tamanho delas deve ser pequeno o suficiente para que a suposição anterior seja verdadeira (os autores utilizaram sub-classes com tamanho em torno de 1s).

Para a estimar as matrizes de variabilidade do LDA, no FLsD utilizamos as sub-classes ao invés das classes, já que não dispomos desta última informação. As notações $\mathbf{S}_b^h$ e $\mathbf{S}_w^h$ serão utilizadas para as matrizes de variabilidade inter e intra-classes, respectivamente. Assim:

$$\mathbf{S}_b^h = \sum_{v \in V} P_v (\bar{\mathbf{x}}_v - \bar{\mathbf{x}})(\bar{\mathbf{x}}_v - \bar{\mathbf{x}})^T \tag{4.8}$$

$$\mathbf{S}_w^h = \sum_{v \in V} P_v E[(\mathbf{X}_v - \bar{\mathbf{x}}_v)(\mathbf{X}_v - \bar{\mathbf{x}}_v)^T] \tag{4.9}$$

onde V é o conjunto de todas as sub-classes e, para todo $v \in V$, existe $h(v) \rightarrow c$.

A ideia principal é que os dados contidos nas sub-classes sejam suficientes para aproximar as matrizes $\mathbf{S}_b^h$ e $\mathbf{S}_w^h$ de $\mathbf{S}_b$ e $\mathbf{S}_w$. Segundo os autores, quanto mais próximas as estatísticas de 1º ordem das sub-classes forem das mesmas estatísticas das classes verdadeiras, então as matrizes de variabilidade das sub-classes tendem a convergir para as matrizes de variabilidade verdadeiras (omitiremos a prova matemática deste trabalho, mas ela encontra-se em (GIANNAKOPOULOS; PETRIDIS, 2012)).

4.3.2 Janelas de Textura

O conceito de Janelas de Textura foi primeiramente utilizado em Tzanetakis e Cook (2002) para capturar informações de longo prazo a partir de dados parametrizados em curto prazo. A ideia é simplesmente calcular as médias e variâncias de cada amostra sobre uma janela iniciada nela. Matematicamente, se temos um conjunto de dados $\mathbf{X}$ em $\mathbb{R}^N$, a janela de textura irá duplicar sua dimensionalidade, sendo as dimensões de 1 a N as médias, e as dimensões de $N+1$ a $2N$ as variâncias. Assim, dada uma janela de tamanho L, podemos dizer que a primeira metade das dimensões do novo conjunto de amostras $\mathbf{X}'$ em $\mathbb{R}^{2N}$ é dada por:

$$\mathbf{x}'_i[n] = \frac{1}{L}\sum_{l=i}^{i+L}\mathbf{x}_l[n] \quad \text{, para todo } n = [1, ..., N] \tag{4.10}$$

e a segunda metade, por:

$$\mathbf{x}'_i[n] = \frac{1}{L}\sum_{l=i}^{i+L}(\mathbf{x}[N-n]_l - \mathbf{x}'_i[N-n])^2 \quad \text{, para todo } n = [N+1, ..., 2N]. \tag{4.11}$$

No FLsD, as janelas de textura foram empregadas sobre o MFCC. Aqui as aplicaremos também sobre o LSP.

4.3.3 Diarização de Locutor

Nesta seção vamos inserir o FLsD em nosso SDL e apresentar os resultados obtidos. Como no artigo de introdução ao FLsD os autores utilizaram o MFCC como técnica de parametrização, e neste trabalho vimos que melhores resultados podem ser conseguidos com o LSP, iremos agora considerar somente estas duas técnicas. Também para clarificar se as janelas de textura exercem ou não alguma modificação importante nos dados, vamos mostrar resultados com e sem sua utilização.

A Tabela 4.4 apresenta o DER para configurações de parametrização com e sem a Janelas de Textura. Observando-a, podemos ver que nem sempre o FLsD consegue resultados melhores com seu uso do que sem. Dos 15 testes apresentados cujos resultados mostraram diferenças estatísticas significantes em relação ao sistema de referência, o FLsD se saiu melhor em 8 deles. Isto é uma margem baixa se considerarmos que é necessário saber qual número de dimensões utilizar.

Outro ponto negativo do FLsD é que ele é ineficaz quando a técnica de parametrização utilizada é o LSP. Apesar do LSP ter sido melhor em todos os resultados sem Janelas de Textura, a diferença para com o Sistema de Referência é insignificante. Entretanto, quando utilizamos o MFCC com nossa configuração padrão (19 coeficientes extraídos de janelas de 30ms a cada 10ms), o FLsD melhora significativamente os resultados em todas as bases exceto a base BNEWS02. Ao contrário do que esperávamos, o uso da mesma configuração de parametrização utilizada no artigo de introdução ao FLsD (MFCC, 12 coeficientes extraídos de janelas de 20ms a cada 20ms) degradou os resultados. Esta observação é muito interessante pois a configuração de 12 coeficientes sozinha obteve melhores resultados no sistema de referência do que a configuração de 19 coeficientes (MFCC). Mas esta tendência não se confirmou com o uso do FLsD.

Com relação as Janelas de Textura, quando à empregamos nos vetores de características, e depois aplicamos o FLsD, o DER apenas melhorou quando a técnica de parametrização utilizada foi o MFCC com 19 coeficientes. Entretanto, para esta mesma

PARAMETRIZAÇÃO	BASE	SIS. REF.	FLsD	DIM.
MFCC, com Janelas de Textura (19,30,10)	**CHOME00**	29.26%	**28.31%**	3
	SWBD00	18.16%	**15.96%**	1
	BNEWS02	**14.22%**	14.64%	17
	SWBD02	15.52%	**13.61%**	1
	AMI_ES	17.73%	**16.47%**	15
LSP, com Janelas de Textura (19,30,10)	**CHOME00**	**23.48%**	24.55%	17
	SWBD00	**12.83%**	13.22%	1
	BNEWS02	**11.71%**	12.41%	18
	SWBD02	**10.65%**	11.71%	13
	AMI_ES	**16.18%**	17.61%	12
MFCC, com Janelas de Textura (12,20,20)	**CHOME00**	**26.90%**	30.05%	3
	SWBD00	16.67%	**16.60%**	1
	BNEWS02	**14.45%**	15.63%	12
	SWBD02	**12.67%**	14.14%	1
	AMI_ES	**17.61%**	19.09%	15
MFCC, sem Janelas de Textura (19,30,10)	**CHOME00**	29.26%	**26.90%**	13
	SWBD00	18.16%	**14.98%**	1
	BNEWS02	**14.22%**	14.24%	18
	SWBD02	15.52%	**13.20%**	1
	AMI_ES	17.73%	**14.64%**	18
LSP, sem Janelas de Textura (19,30,10)	**CHOME00**	23.48%	**22.99%**	17
	SWBD00	12.83%	**12.57%**	16
	BNEWS02	11.71%	**11.36%**	16
	SWBD02	10.65%	**10.32%**	17
	AMI_ES	16.18%	**16.16%**	15

Tabela 4.4: Resultados de DER para dados com e sem janelas de textura para sub-classes de tamanho igual a 1s. A primeira coluna mostra como foi feita a parametrização, sendo os 3 números entre parênteses o número de coeficientes, tamanho da janela e espaçamento (em ms) dos dados de curto prazo. A terceira e quarta coluna mostram o DER obtido pelo sistema de referência (sem Janelas de Textura) e o pelo sistema com FLsD, respectivamente. Resultados sublinhados indicam que não há diferença estatística entre eles. A coluna 5 mostra em qual dimensão foi observado o melhor DER (coluna 4).

parametrização, a aplicação do FLsD sem Janelas de Textura obteve resultados ainda melhores. Quando a técnica de parametrização foi o LSP ou o MFCC com 12 coeficientes, a utilização de Janelas de Textura degradou ou não melhorou os resultados.

4.4 Melhoramentos ao FLsD

Nesta seção iremos explorar um pouco mais o FLsD. Acreditamos que o conceito de sub-classes ainda pode ser melhorado, e por isso iremos propor a utilização de diferentes tipos de sub-classes, baseadas na saída do nosso algoritmo de segmentação. Também testaremos o comportamento do FLsD quando submetido a combinações de técnicas de parametrização de diferentes tipos, com base nos bons resultados encontrados na Seção 4.2.

4.4.1 Pré-Segmentação

Sobre a suposição dada na Seção 4.3.1: "quando analisamos um sinal de áudio referente a uma conversa entre pessoas, podemos pressupor que, para cada amostra, todos os seus vizinhos dentro de uma janela relativamente pequena, irão pertencer ao mesmo locutor". Apesar desta suposição estar correta na maior parte do tempo, ela pode falhar por dois motivos: (1) a amostra selecionada pode encontrar-se justamente na divisa entre o término da fala de um locutor e o inicio de outro, ou (2) ela encontra-se em uma região onde há sobreposição de fala. Desta forma, quanto menor for o tamanho da janela em torno da amostra selecionada, menor será o risco de contaminação. Em contra partida, uma janela pequena implica em poucos dados.

Nossa hipótese é que a utilização de segmentos gerados através de um segmentador por Janelas Deslizantes, possa ser uma alternativa melhor para a criação das sub-classes do que simplesmente seccionar o áudio em partes de tamanho fixo. Analisando as configurações de segmentação apresentadas anteriormente, na Seção 2.4, selecionamos 4 delas que possuem uma boa relação de tamanho e pureza (quanto mais melhor). São elas as configurações 5, 10, 13 e 18 (ver Tabela 3.4).

A simulação foi dividida em 3 partes. A primeira e a segunda parte consistem em comparar o desempenho do FLsD com sub-classes "de tamanho fixo" vs. "com segmentação", e sem o uso de janelas de textura. Para isso, utilizamos como técnicas de parametrização o MFCC (primeira parte) e o LSP (segunda parte), ambos com 19 coeficientes, extraídos sobre janelas de 30ms a cada 10ms. E na terceira parte empregamos o uso de Janelas de Textura juntamente com MFCC.

A Tabela 4.5 apresenta os menores DERs encontrados dentre todas as 19 dimensões possíveis para cada base. A primeira linha de cada parte representa nossa referência, que, para estes testes, foram os resultados do FLsD com sub-classes de tamanho fixo, extraídas sequencialmente a cada 1 segundo. Os valores sublinhados são aqueles onde não há significância estatística ($\alpha = 0.05$) para com o DER obtido pela referência.

Observando estes resultados, notamos que na primeira parte, a DL com sub-classes dadas por segmentação não apresentaram nenhuma melhora significante nas bases CHOME00, SWBD00 e SWBD02, e ainda houve um aumento no DER na base AMI_ES. Entretanto houve uma melhora relativa de $\sim$7.5% na base BNEWS02 quando utilizamos a segmentação de ID 10. Observando os DERs na base AMI_ES é possível perceber que eles melhoram a medida que o tamanho dos segmentos aumenta, ao ponto da configuração 18 poder ser considerada indiferente das sub-classes de tamanho fixo.

Na segunda parte, o uso de sub-classes dadas por segmentação não melhora (e nem

PARAM.	JT	SC	DER				
			CHOME00	SWBD00	BNEWS02	SWBD02	AMI_ES
MFCC	não	a cada 1s	26.90%	14.98%	14.24%	13.20%	**14.64%**
MFCC	não	seg ID 5	27.12%	15.14%	13.23%	13.32%	16.25%
MFCC	não	seg ID 10	26.91%	14.94%	**13.18%**	13.73%	16.18%
MFCC	não	seg ID 13	**26.52%**	**14.64%**	13.86%	**12.95%**	15.81%
MFCC	não	seg ID 18	26.88%	14.91%	13.30%	13.16%	15.00%
LSP	não	a cada 1s	22.99%	**12.57%**	11.36%	**10.32%**	**16.16%**
LSP	não	seg ID 5	**22.55%**	12.69%	**10.94%**	10.47%	16.44%
LSP	não	seg ID 10	22.87%	12.76%	11.13%	10.26%	16.44%
LSP	não	seg ID 13	22.83%	12.77%	10.99%	10.59%	16.44%
LSP	não	seg ID 18	23.15%	12.84%	11.45%	10.66%	16.44%
MFCC	sim	a cada 1s	28.31%	15.96%	**14.64%**	13.61%	**16.47%**
MFCC	sim	seg ID 5	28.17%	16.18%	15.10%	**13.22%**	17.45%
MFCC	sim	seg ID 10	27.62%	17.59%	14.35%	14.14%	17.80%
MFCC	sim	seg ID 13	**26.04%**	**15.34%**	15.85%	13.30%	17.54%
MFCC	sim	seg ID 18	28.24%	17.41%	14.51%	14.87%	17.24%

Tabela 4.5: DER para vários tipos de sub-classes, baseadas nas configurações do Algoritmo de Janelas Deslizantes. (JT - Janelas de Textura, SC - Sub-Classes).

degrada) em nada os resultados de DER com o LSP. Este teste, somado aos resultados dos testes anteriores, nos diz que a aplicação do FLsD sobre o LSP em nada contribui para que haja alguma melhora nos resultados. Assim podemos concluir com segurança que o LSP não é uma técnica de parametrização adequada para ser utilizada com FLsD.

Por último, no terceiro teste, podemos ver parcialmente que a utilização de sub-classes por segmentação contribui para a melhora dos resultados quando aplicamos a técnica de Janelas de Textura no MFCC. Nas bases CHOME00 e SWBD00 a redução no DER foi significante quando utilizamos as segmentações de ID 10 e 13, com uma pequena vantagem para a segmentação 13. Entretanto a segmentação 13 gera piores resultados na base BNEWS02 do que a segmentação 10. O aumento do DER novamente aconteceu na base AMI_ES, e novamente com uma ligeira tendência a melhorar a medida que se aumente o tamanho dos segmentos.

4.4.2 Múltiplas Parametrizações

Deve-se ressaltar que cada uma das técnicas de parametrização mostradas neste trabalho foi projetada de maneira diferente umas das outras, visando coletar diferentes informações sobre o áudio. Nesta seção, trabalharemos com a hipótese de que a junção de duas técnicas diferentes de parametrização, com aplicação de LDA ou FLsD, pode acarretar em um melhor desempenho na tarefa de DL.

4.4.2.1 Aplicação do LDA

Primeiramente, iremos aplicar o LDA sobre todas as combinações, dois a dois, de parametrizações com MFCC, MEL, LSP e LPCC. Para que a concatenação das matrizes de características fosse possível, todas estas técnicas foram aplicadas com a mesma configuração de tamanho de janela e espaçamento. Aqui utilizamos 19 coeficientes, com janelas de 30ms espaçadas por 10ms. Assim, o número de vetores característica extraídos é igual em todos os testes, e estão alinhados. Como cada linha da matriz de caracterís-

ticas representa um vetor de característica, a concatenação efetuada foi horizontal, ou seja, linha a linha, de forma que a dimensionalidade final dobrasse de tamanho, para 38 (19+19). Nesta seção não utilizaremos janelas de textura, pois isto aumentaria ainda mais o número de dimensões - saltando de 38 para 76 - e podendo acarretar no cálculo de matrizes de covariância com o número de dimensões quase igual ao número de vetores característica em segmentos pequenos.

A Tabela 4.6 mostra os resultados de DER após a aplicação do LDA sobre diversas combinações de parâmetros. A dimensionalidade final em cada gravação foi fixada em $L-1$, onde L é o número de locutores do áudio sob avaliação. Note que $L-1$ é também a dimensionalidade ideal para o LDA, como foi visto anteriormente na Seção 4.2. Neste teste, estamos apenas explorando o potencial que nossa hipótese possui, dado que o LDA não é diretamente aplicável a DL.

LDA	**PARAM.**	**DER**				
		CHOME00	**SWBD00**	**BNEWS02**	**SWBD02**	**AMI_ES**
SIM	LSP+LPCC	21.14%	**10.65%**	7.93%	8.87%	11.86%
	LSP+MEL	21.02%	10.96%	7.31%	**8.68%**	11.54%
	MEL+LPCC	21.20%	11.07%	8.39%	9.29%	12.16%
	MFCC+LPCC	**20.54%**	10.83%	7.30%	9.32%	**11.22%**
	MFCC+LSP	21.20%	10.78%	**7.20%**	8.84%	11.74%
	MFCC+MEL	21.85%	11.33%	8.23%	9.67%	14.79%
	LPCC	22.39%	10.91%	9.01%	9.95%	16.23%
	LSP	22.25%	10.91%	9.01%	9.36%	16.00%
	MEL	21.99%	12.01%	9.74%	10.59%	18.01%
	MFCC	22.39%	11.75%	9.64%	9.68%	15.66%
NÃO	LSP+LPCC	24.06%	12.97%	11.73%	**10.62%**	18.51%
	LSP+MEL	25.22%	14.76%	13.21%	12.35%	18.23%
	MEL+LPCC	28.49%	16.57%	12.68%	13.78%	16.90%
	MFCC+LPCC	27.73%	16.51%	13.77%	14.01%	16.74%
	MFCC+LSP	26.60%	14.96%	12.24%	12.86%	17.21%
	MFCC+MEL	28.71%	18.21%	12.45%	14.83%	18.14%
	LPCC	**23.41%**	13.55%	12.51%	11.67%	**15.30%**
	LSP	23.48%	**12.83%**	**11.71%**	10.65%	16.18%
	MEL	29.14%	17.71%	13.02%	15.60%	17.67%
	MFCC	29.26%	18.16%	14.22%	15.52%	17.73%

Tabela 4.6: DER com e sem aplicação do LDA para dados de simples ou múltiplas parametrizações.

Analisando os resultados apresentados na Tabela 4.6, podemos ver que o uso da LDA novamente apresenta bons ganhos de desempenho. Em todos os testes, o menor valor DER com LDA foi significantemente menor que o menor valor sem aplicação do LDA. Nas bases SWDB00, SWBD02 e AMI_ES onde o número de locutores é fixo, o número de dimensões utilizadas foi de 1,1 e 3, respectivamente. Outra rápida observação que pode ser feita é que, quando utilizamos o LDA, os melhores resultados são obtidos com combinações de técnicas de parametrização, enquanto que, quando não utilizamos LDA, os melhores resultados ocorrem sem a utilização de múltiplas parametrizações. Acreditamos que, neste último caso, a utilização de múltiplas técnicas de parametrização deteriora

os resultados principalmente devido ao grande número de parâmetros necessários para se estimar um modelo de 38 dimensões.

Os resultados com o uso do LDA contribuem fortemente com nossa hipótese de que múltiplas parametrizações do sinal, submetidas à análise discriminante, melhoram o processo de agrupamento. Isto pode ser visto observando a parte superior da tabela, onde a grande maioria dos resultados de LDA com múltiplas parametrizações, são melhores que os resultados com parametrizações simples. Outra importante observação é que, com exceção da base SWBD00, os melhores resultados são oriundos da junção de técnicas de coeficientes espectrais (MFCC e MEL) com técnicas de predição linear (LPCC e LSP). A junção do MFCC com MEL não gera ganhos significativos. E a junção do LPCC e LSP, as duas técnicas que obtiveram melhores resultados individualmente, não produziu resultados tão bons como se era esperado, principalmente na base AMI_ES.

Estes resultados apontam que a junção de técnicas de parametrização de naturezas diferentes pode ajudar no desempenho de um SDL, principalmente quando submetidas à análise discriminante.

4.4.2.2 Aplicação do FLsD

Agora utilizaremos o FLsD no mesmo cenário anterior de múltiplas parametrizações. Como foi visto, o FLsD é mais adequado à DL do que o LDA, pois não precisa de dados previamente rotulados. Entretanto seus resultados não são tão bons quanto o LDA.

Aplicamos o FLsD sobre as várias configurações de parametrização considerando dois tipos de sub-classes: as de tamanho fixado em 1s, como na formulação original, e as de tamanho variado, de acordo com a segmentação de configuração 13. O uso desta configuração é justificado na Seção 4.4.1, onde vimos que ela apresentou resultados melhores do que as demais na maioria das bases testadas. Nesta seção não utilizaremos Janelas de Textura pela mesma razão de alta dimensionalidade comentada na seção anterior.

Os resultados obtidos são apresentados na Tabela 4.7. Quando utilizamos FLsD, tanto com sub-classes de tamanho fixo (1s), quanto de tamanho variável (seg. 13), os melhores resultados foram obtidos com a técnica LSP ou a sua junção com o LPCC nas três bases cujas gravações são feitas por meio de linhas telefônicas (CHOME00, SWBD00 e SWBD02). É interessante notar que resultados semelhantes também são vistos com o uso do LDA na Tabela 4.6, o que nos leva a crer que o LSP é uma técnica mais robusta para este tipo de gravação.

As técnicas baseadas no domínio da frequência apresentaram uma vantagem mais clara nas outras duas bases (BNEWS02 e AMI_ES), cujas gravações foram feitas com equipamentos profissionais (transmissões de rádio e televisão, e reuniões). Elas obtiveram os melhores resultados quando somadas ao LSP ou ao LPCC. Entretanto, uma melhora significativa apenas foi observada para o MFCC+LSP na base AMI_ES com sub-classes de 1s. Nas demais, a diferença não foi significante com relação ao LSP+LPCC.

Podemos concluir que na maioria dos casos, o uso do FLsD com múltiplas técnicas de parametrização consegue obter resultados melhores do que quando utilizados com apenas uma destas técnicas. Mas há uma restrição: a melhora só é observada quando ao menos uma delas é baseada em Predição Linear.

Outro fato interessante que podemos inferir observando a Tabela 4.7 é que o uso de sub-classes com segmentação, quando comparado às de tamanho fixo de 1s, gerou resultados significantemente melhores nas bases CHOME00 e AMI_ES.

SUB-CLAS.	PARAM.	DER				
		CHOME00	SWBD00	BNEWS02	SWBD02	AMI_ES
a cada 1s	MFCC	28.03%	14.89%	16.94%	13.46%	24.55%
a cada 1s	MEL	27.32%	15.97%	15.79%	14.91%	28.15%
a cada 1s	LSP	25.75%	13.21%	13.78%	12.19%	26.73%
a cada 1s	LPCC	27.01%	13.10%	14.36%	12.46%	25.28%
a cada 1s	MFCC+LSP	27.18%	13.13%	13.76%	12.56%	**21.33%**
a cada 1s	MFCC+MEL	28.04%	14.79%	15.90%	13.30%	27.21%
a cada 1s	MFCC+LPCC	26.30%	13.08%	14.18%	12.58%	22.09%
a cada 1s	LSP+MEL	25.98%	13.86%	**13.27%**	11.69%	24.54%
a cada 1s	LSP+LPCC	**25.27%**	**12.43%**	14.74%	**11.28%**	22.47%
a cada 1s	MEL+LPCC	26.75%	13.63%	14.20%	12.24%	24.89%
seg. ID 13	MFCC	26.98%	14.66%	15.95%	13.21%	23.19%
seg. ID 13	MEL	26.86%	15.93%	16.36%	13.76%	28.28%
seg. ID 13	LSP	24.40%	13.21%	13.64%	12.10%	21.91%
seg. ID 13	LPCC	25.35%	13.30%	14.92%	11.97%	22.87%
seg. ID 13	MFCC+LSP	24.70%	12.87%	13.54%	11.97%	**18.07%**
seg. ID 13	MFCC+MEL	26.72%	14.78%	14.68%	12.06%	23.66%
seg. ID 13	MFCC+LPCC	25.27%	13.02%	13.95%	12.05%	19.60%
seg. ID 13	LSP+MEL	24.64%	13.42%	**13.48%**	**11.59%**	21.17%
seg. ID 13	LSP+LPCC	**24.33%**	**12.49%**	13.70%	11.73%	18.14%
seg. ID 13	MEL+LPCC	24.88%	13.47%	13.94%	12.25%	19.74%

Tabela 4.7: Aplicação do FLsD com sub-classes de tamanho fixo ou variável (de acordo com a segmentação 13) e com simples ou múltiplas parametrizações.

5 CONCLUSÃO E TRABALHOS FUTUROS

Neste capítulo iremos fazer as considerações finais sobre este trabalho. Na Seção 5.1 apresentaremos as conclusões finais, tanto sobre o Sistema de Referência, quanto sobre a aplicação das técnicas de redução de dimensionalidade. E na Seção 5.2, falaremos das perspectivas para trabalhos futuros.

5.1 Conclusão

Este trabalho tratou, inicialmente, da construção de um Sistema de Diarização de Locutor. Foram empregados métodos e algoritmos já consolidados na área, que realizam as tarefas de parametrização (e.g. MFCC, MEL, LSP e LPCC), segmentação (e.g. Algoritmo de Janelas Deslizantes) e agrupamento de locutor (e.g. Agrupamento Hierárquico Aglomerativo). Depois, métodos de redução de dimensionalidade baseados em análises estatísticas foram empregados, no intuito de se tentar reduzir o tempo de execução gasto pelo AHC, e melhorar o desempenho geral do SDL.

As próximas seções descrevem detalhadamente as conclusões para estas duas etapas do trabalho.

5.1.1 Sistema de Referência

Com relação à parametrização do sinal, constatamos que métodos baseados em predição linear, como o LSP e o LPCC, alcançaram melhores resultados de DER. Já os métodos MEL e MFCC (este último amplamente utilizado) baseados na Análise de Fourier, obtiveram resultados aquém do esperado, sendo significativamente piores do que os métodos de predição linear, em todas as 5 bases utilizadas.

Na segmentação de locutor, utilizamos o algoritmo de Janelas Deslizantes, e observamos os efeitos que seus parâmetros causam sobre a segmentação final. As medidas, feitas em termos de pureza de *cluster*, constataram que configurações que prezam pela pureza, geram segmentos menores, e as configurações que prezam por segmentos maiores, geram segmentos mais impuros. A boa relação entre pureza e tamanho dos segmentos é necessária para se obter um bom desempenho de agrupamento. Nossos resultados também apontaram uma forte relação com o trabalho de Kotti *et al* (2008), pois, a configuração de segmentação que obteve os melhores resultados de DER, é também aquela cuja distribuição Gaussiana sobre o tamanho dos segmentos gerados, é mais próxima da segmentação "real" mostrada pelos autores.

No módulo de agrupamento, testamos no método AHC todas as configurações de parametrização, segmentação, medidas de distância e critérios de parada. Vimos que a distância d_{GLR} é sem dúvida a mais apropriada a esta tarefa. E que, mais importante do

que a média, é a variância dos dados. Pois a distância d_{GLR_Σ} obteve os mesmos resultados de DER, porém com um menor tempo de execução.

Detalhamos o comportamento do critério de parada para as distâncias ΔBIC e d_{GLR_Σ}. Aqui constatamos que, embora o valor λ do ΔBIC seja mais simples de se ajustar, regular um limiar d_{GLR_Σ} gera melhores resultados. Também observamos que a base AMI_ES, por conter áudios mais longos, gerou um comportamento totalmente diferente das outras bases. Nela, o ΔBIC com gaussianas simples não é capaz de discriminar locutores a partir de *clusters* muito grandes, sendo o melhor resultado com DER acima de 40%. No caso do d_{GLR_Σ}, o valor do melhor limiar está bem acima dos melhores limiares para as bases do NIST (14000 contra 1900). Este comportamento já era esperado pois, segundo Han *et al* (2008), o d_{GLR} tende a crescer exponencialmente com o aumento da quantidade de dados.

Apresentamos uma forma de reduzir o tempo de execução do módulo de agrupamento para a distância d_{GLR_Σ}, sem que haja perdas de desempenho. Nesta técnica, calculamos a matriz de covariância referente à união dos dados, a partir das matrizes individuais previamente calculadas.

5.1.2 Redução de Dimensionalidade

Com relação à redução de dimensionalidade, aplicamos 3 técnicas diretamente sobre os vetores de características: a Análise de Componentes Principais, a Análise de Discriminantes Lineares e a Análise de Semi-Discriminantes Lineares. A análise que obteve os melhores resultados com a menor quantidade de dimensões foi o LDA. No entanto, esta exige que existam os rótulos do áudio para a computação da matriz de transformação, e sabemos que esta informação não está disponível em DL. O PCA, que atua de maneira não-supervisionada, não obteve melhoras de desempenho, mas conseguiu reduzir o número de dimensões de 19 para 12 sem perdas. Esta redução acarreta também em menor uso de memória (consumida pelos dados) e menor tempo de execução (de algoritmos de Aprendizado de Máquina).

O FLsD, que é um método para redução de dimensionalidade não-supervisionado, baseado na LDA, foi ostensivamente testado. Podemos inferir que seu uso, quando aplicado sobre o MFCC, realmente obtêm um melhor desempenho do que o PCA. Mas quando aplicado sobre o LSP, o desempenho permanece igual em relação ao uso do PCA ou aos dados sem aplicação de nenhuma técnica. Mesmo quando não há vantagem de desempenho sobre o PCA, o FLsD consegue reduzir os dados para um número menor de dimensões do que o PCA.

Também analisamos o uso de Janelas de Textura. Esta técnica de suavização foi utilizada sobre os dados no artigo de apresentação do FLsD. Notamos que o seu uso não acarreta melhoras no desempenho, e, em alguns casos, acaba por degradá-lo.

Por fim, propusemos duas alterações ao método FLsD para tentar melhorar seu desempenho. A primeira foi o uso da saída do segmentador para substituir as sub-classes, no intuito de estimar melhores matrizes intra e inter classes. De modo geral, melhores resultados foram obtidos nas bases do NIST. Entretanto, poucos foram significantemente melhores. Na base AMI_ES todos os resultados deste método foram piores do que o uso das sub-classes de tamanho fixo.

A segunda alteração foi a utilização de múltiplas técnicas de parametrização do sinal, com aplicação de redução de dimensionalidade. Visto que cada técnica de parametrização foi projetada para capturar diferentes informações do sinal, partimos da hipótese que a junção delas poderia contribuir no processo de discriminação de locutores. Para testar

esta hipótese, inicialmente aplicamos apenas o LDA sobre todas as combinações 2 a 2 de técnicas de parametrização utilizadas neste trabalho. Constatamos com isso que a junção de fato melhora o desempenho final. E notamos que os pares de parametrizações devem contar com ao menos uma técnica de predição linear. Percebemos que melhores resultados são alcançados quando se utiliza uma técnica de predição linear combinada com outra baseada no domínio da frequência. Quando não aplicamos o LDA, juntar técnicas de parametrização é prejudicial, pois eleva demasiadamente o número de dimensões dos dados e acaba por degradar o desempenho.

Quando submetemos as múltiplas técnicas de parametrização ao FLsD, os resultados também foram melhores em 3 bases: SWBD00, SWBD02 e AMI_ES. Nas outras duas eles não superaram significantemente o LSP sozinho. Interessantemente, os resultados na base AMI_ES foram ainda melhores quando utilizamos o FLsD com duas parametrizações e sub-classes de tamanho variável, chegando a 15.28% de melhora relativa sobre o uso de sub-classes de tamanho fixo.

5.2 Trabalhos Futuros

Pretendemos investigar, utilizando diversos algoritmos de segmentação de locutor, se é possível obter a melhor configuração de parâmetros para estes algoritmos, apenas comparando a distribuição Gaussiana dos segmentos gerados. Assim, as melhores configurações seriam aquelas que gerassem uma saída mais próxima da distribuição real, como constatamos com o algoritmo de Janelas Deslizantes, durante a Seção 3.3.1.

Também, durante todo o trabalho, modelamos os locutores com gaussianas simples. Uma próxima etapa seria a utilização de modelos de mistura para este fim. Esperamos que estes modelos mais complexos ajudem a melhorar principalmente o desempenho na base AMI_ES, que possui áudios maiores, e que acabam gerando *cluters* muito grandes para se modelar com apenas uma gaussiana.

Como a redução de dimensionalidade acarreta em dados mais compactos, e consequentemente em gaussianas com menos parâmetros, poderíamos compensar esta diferença na quantidade de parâmetros, aumentando o número de gaussianas do modelo.

Planejamos também evoluir o SDL atual para ser capaz de realizar a segmentação e agrupamento de locutor em passo único, através de iterações sobre o Algoritmo de Viterbi. Desta forma, poderemos testar o desempenho dos novos vetores de características em um sistema mais próximo daqueles no estado-da-arte.

Com esta evolução no sistema, tentaremos nos inserir como participantes de algumas das avaliações realizadas pelo NIST. Seria uma atitude pioneira no Brasil, pois até hoje nenhuma equipe brasileira atuou nas avaliações anteriores.

REFERÊNCIAS

AJMERA, J.; WOOTERS, C. A robust speaker clustering algorithm. In: AUTOMATIC SPEECH RECOGNITION AND UNDERSTANDING, 2003. ASRU'03. 2003 IEEE WORKSHOP ON. **Anais...** [S.l.: s.n.], 2003. p.411–416.

ANGUERA MIRO, X. et al. Speaker Diarization: a review of recent research. **Audio, Speech, and Language Processing, IEEE Transactions on**, [S.l.], v.20, n.2, p.356 – 370, feb. 2012.

ANGUERA, X.; WOOTERS, C.; HERNANDO, J. Friends and enemies: a novel initialization for speaker diarization. In: INTERSPEECH. **Anais...** [S.l.: s.n.], 2006.

BUNDY, A.; WALLEN, L. Linear Predictive Coding. In: CATALOGUE OF ARTIFICIAL INTELLIGENCE TOOLS. **Anais...** Springer Berlin Heidelberg, 1984. p.61–61. (Symbolic Computation).

CAMPBELL J.P., J. Speaker recognition: a tutorial. **Proceedings of the IEEE**, [S.l.], v.85, n.9, p.1437–1462, 1997.

CHEN, S. S.; GOPALAKRISHNAN, P. S. Speaker, Environment And Channel Change Detection And Clustering Via The Bayesian Information Criterion. In: **Anais...** [S.l.: s.n.], 1998. p.127–132.

CHENG, S.-S.; WANG, H.-M.; FU, H.-C. BIC-based speaker segmentation using divide-and-conquer strategies with application to speaker diarization. **Audio, Speech, and Language Processing, IEEE Transactions on**, [S.l.], v.18, n.1, p.141–157, 2010.

DELACOURT, P.; KRYZE, D.; WELLEKENS, C. J. DISTBIC: a speaker-based segmentation for audio data indexing. In: SPEECH COMMUNICATION. **Anais...** [S.l.: s.n.], 2000. p.111–126.

DUDA, R. O.; HART, P. E.; STORK, D. G. **Pattern classification**. [S.l.]: John Wiley & Sons, 2012.

FREDOUILLE, C.; BOZONNET, S.; EVANS, N. The LIA-EURECOM RT '09 Speaker Diarization System. In: RT'09, NIST RICH TRANSCRIPTION WORKSHOP, MAY 28-29, 2009, MELBOURNE, FLORIDA, USA. **Anais...** [S.l.: s.n.], 2009.

FREDOUILLE, C.; EVANS, N. The LIA RT'07 Speaker Diarization System. In: MULTIMODAL TECHNOLOGIES FOR PERCEPTION OF HUMANS. **Anais...** Springer Berlin Heidelberg, 2008. p.520–532. (Lecture Notes in Computer Science, v.4625).

FRIEDLAND, G. et al. Prosodic and other Long-Term Features for Speaker Diarization. **Audio, Speech, and Language Processing, IEEE Transactions on**, [S.l.], v.17, n.5, p.985 –993, july 2009.

FUKUNAGA, K. **Introduction to Statistical Pattern Recognition**. 2.ed. [S.l.]: Academic Press, 1990.

FURUI, S. et al. Introduction to the Special Section on New Frontiers in Rich Transcription. **IEEE Transactions on Audio, Speech, and Language Processing**, [S.l.], v.20, n.2, p.353–355, 2012.

GAUVAIN, J.-L.; LAMEL, L.; ADDA, G. Partitioning and transcription of broadcast news data. In: ICSLP. **Anais...** [S.l.: s.n.], 1998. v.98, n.5, p.1335–1338.

GIANNAKOPOULOS, T.; PETRIDIS, S. Fisher Linear Semi-Discriminant Analysis for Speaker Diarization. **Audio, Speech, and Language Processing, IEEE Transactions on**, [S.l.], v.20, n.7, p.1913 –1922, sept. 2012.

GISH, H.; SCHMIDT, M. Text-independent speaker identification. **Signal Processing Magazine, IEEE**, [S.l.], v.11, n.4, p.18–32, 1994.

GISH, H.; SIU, M.-H.; ROHLICEK, R. Segregation of speakers for speech recognition and speaker identification. In: ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, 1991. ICASSP-91., 1991 INTERNATIONAL CONFERENCE ON. **Anais...** [S.l.: s.n.], 1991. p.873–876 vol.2.

HACHEM, K. et al. Robust Unsupervised Speaker Segmentation for Audio Diarization. **Signal Processing**, [S.l.], p.307–320, 2010.

HAN, K. J.; NARAYANAN, S. S. A robust stopping criterion for agglomerative hierarchical clustering in a speaker diarization system. In: INTERSPEECH. **Proceedings...** [S.l.: s.n.], 2007. p.1853–1856.

HAN, K. J.; NARAYANAN, S. S. Agglomerative hierarchical speaker clustering using incremental Gaussian mixture cluster modeling. **Proc. Interspeech'08, Brisbane, Australia**, [S.l.], p.20–23, 2008.

HASTIE, T. et al. The elements of statistical learning: data mining, inference and prediction. **The Mathematical Intelligencer**, [S.l.], v.27, n.2, p.83–85, 2005.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. J. H. **The elements of statistical learning**. [S.l.]: Springer New York, 2001. v.1.

HERMANSKY, H.; HANSON, B.; WAKITA, H. Perceptually based linear predictive analysis of speech. In: ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, IEEE INTERNATIONAL CONFERENCE ON ICASSP '85. **Anais...** [S.l.: s.n.], 1985. v.10, p.509–512.

HUANG, X. et al. **Spoken language processing**. [S.l.]: Prentice Hall PTR New Jersey, 2001. v.15.

IMSENG, D.; FRIEDLAND, G. Robust speaker diarization for short speech recordings. In: AUTOMATIC SPEECH RECOGNITION & UNDERSTANDING, 2009. ASRU 2009. IEEE WORKSHOP ON. **Anais...** [S.l.: s.n.], 2009. p.432–437.

KADRI, H. et al. Hybrid approach for unsupervised audio speaker segmentation. **Proceedings of the EURASIP EUSIPCO'06**, [S.l.], 2006.

KOTTI, M.; BENETOS, E.; KOTROPOULOS, C. Computationally efficient and robust BIC-based speaker segmentation. **Audio, Speech, and Language Processing, IEEE Transactions on**, [S.l.], v.16, n.5, p.920–933, 2008.

LEEUWEN, D.; KONEčNý, M. Progress in the AMIDA Speaker Diarization System for Meeting Data. In: MULTIMODAL TECHNOLOGIES FOR PERCEPTION OF HUMANS. **Anais...** Springer Berlin Heidelberg, 2008. p.475–483. (Lecture Notes in Computer Science, v.4625).

LI, H.; MA, B.; LEE, K. Spoken Language Recognition: from fundamentals to practice. **Proceedings of the IEEE**, [S.l.], v.101, n.5, p.1136–1159, 2013.

MERMELSTEIN, P. Distance measures for speech recognition, psychological and instrumental. **Pattern recognition and artificial intelligence**, [S.l.], v.116, p.374–388, 1976.

MIRKIN, B. G. **Mathematical classification and clustering**. [S.l.]: Kluwer Academic Pub, 1996. v.11.

MIRO, X. A. **Robust speaker diarization for meetings**. 2006. Tese (Doutorado em Ciência da Computação) — Universitat Politècnica de Catalunya.

PARDO, J.; ANGUERA, X.; WOOTERS, C. Speaker diarization for multiple-distant-microphone meetings using several sources of information. **Computers, IEEE Transactions on**, [S.l.], v.56, n.9, p.1212–1224, 2007.

PICONE, J. W. Signal modeling techniques in speech recognition. In: PROCEEDINGS OF THE IEEE. **Anais...** [S.l.: s.n.], 1993. p.1215–1247.

RABINER, L. A tutorial on hidden Markov models and selected applications in speech recognition. **Proceedings of the IEEE**, [S.l.], v.77, n.2, p.257–286, 1989.

RABINER, L.; JUANG, B.-H. **Fundamentals of speech recognition**. [S.l.]: Prentice hall, 1993.

RABINER, L. R.; SCHAFER, R. W. **Digital processing of speech signals**. [S.l.]: IET, 1979. v.19.

RAMIREZ, J.; GÓRRIZ, J. M.; SEGURA, J. C. Voice activity detection. fundamentals and speech recognition system robustness. **Robust Speech Recognition and Understanding**, [S.l.], p.1–22, 2007.

RAMíREZ, J. et al. SVM-Enabled Voice Activity Detection. In: ADVANCES IN NEURAL NETWORKS - ISNN 2006. **Anais...** Springer Berlin Heidelberg, 2006. p.676–681. (Lecture Notes in Computer Science, v.3972).

REYNOLDS, D. A.; QUATIERI, T. F.; DUNN, R. B. Speaker verification using Adapted Gaussian mixture models. In: DIGITAL SIGNAL PROCESSING. **Anais...** [S.l.: s.n.], 2000. p.19–41.

SHEN, J.-l.; HUNG, J.-w.; LEE, L.-s. Robust entropy-based endpoint detection for speech recognition in noisy environments. In: ICSLP. **Proceedings...** [S.l.: s.n.], 1998. v.98.

SIEGLER, M. A. et al. Automatic segmentation, classification and clustering of broadcast news audio. In: DARPA BROADCAST NEWS WORKSHOP. **Proceedings...** [S.l.: s.n.], 1997. p.11.

STAFYLAKIS, T.; KATSOUROS, V. A Review of Recent Advances in Speaker Diarization with Bayesian Methods. In: SPEECH AND LANGUAGE TECHNOLOGIES. **Anais...** InTech, 2011. p.217–240.

STANDARDS, N. I. of; (NIST), T. **2000 Speaker Recognition Evaluation Plan**. Online; acessado em 22-august-2012, Disponível: http://www.itl.nist.gov/iad/mig/tests/spk/2000/spk-2000-plan-v1.0.htm.

STANDARDS, N. I. of; (NIST), T. **2002 Speaker Recognition Evaluation Plan**. Online; acessado em 22-august-2012, Disponível: http://www.itl.nist.gov/iad/mig/tests/spk/2002/.

STANDARDS, N. I. of; (NIST), T. **Spring 2006 (RT-06S) Rich Transcription Meeting Recognition Evaluation Plan**. Online; acessado em 22-august-2012, Disponível: http://www.itl.nist.gov/iad/mig/tests/rt/2006-spring/index.html.

TANG, H. et al. Partially Supervised Speaker Clustering. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, [S.l.], v.34, n.5, p.959–971, May 2012.

TEMKO, A.; MACHO, D.; NADEU, C. Enhanced SVM Training for Robust Speech Activity Detection. In: ACOUSTICS, SPEECH AND SIGNAL PROCESSING, 2007. ICASSP 2007. IEEE INTERNATIONAL CONFERENCE ON. **Anais...** [S.l.: s.n.], 2007. v.4, p.IV–1025–IV–1028.

TRANTER, S.; REYNOLDS, D. An overview of automatic speaker diarization systems. **Audio, Speech, and Language Processing, IEEE Transactions on**, [S.l.], v.14, n.5, p.1557–1565, Sept. 2006.

TRITSCHLER, A.; GOPINATH, R. A. Improved speaker segmentation and segments clustering using the bayesian information criterion. In: EUROSPEECH. **Anais...** [S.l.: s.n.], 1999. v.99, p.679–682.

TZANETAKIS, G.; COOK, P. Musical genre classification of audio signals. **Speech and Audio Processing, IEEE Transactions on**, [S.l.], v.10, n.5, p.293 – 302, jul 2002.

VIJAYASENAN, D.; VALENTE, F.; BOURLARD, H. An information theoretic approach to speaker diarization of meeting data. **Audio, Speech, and Language Processing, IEEE Transactions on**, [S.l.], v.17, n.7, p.1382–1393, 2009.

WILCOX, L. et al. Segmentation of speech using speaker identification. In: ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, 1994. ICASSP-94., 1994 IEEE INTERNATIONAL CONFERENCE ON. **Anais...** [S.l.: s.n.], 1994. v.i, p.I/161–I/164 vol.1.

WOOTERS, C.; HUIJBREGTS, M. The ICSI RT07s speaker diarization system. In: **Multimodal Technologies for Perception of Humans**. [S.l.]: Springer, 2008. p.509–519.

YOU, C. H.; LEE, K.-A.; LI, H. A GMM supervector Kernel with the Bhattacharyya distance for SVM based speaker recognition. In: ACOUSTICS, SPEECH AND SIGNAL PROCESSING, 2009. ICASSP 2009. IEEE INTERNATIONAL CONFERENCE ON. **Anais...** [S.l.: s.n.], 2009. p.4221–4224.

ZHU, X. et al. Multi-stage Speaker Diarization for Conference and Lecture Meetings. In: MULTIMODAL TECHNOLOGIES FOR PERCEPTION OF HUMANS. **Anais...** Springer Berlin Heidelberg, 2008. p.533–542. (Lecture Notes in Computer Science, v.4625).

ZWEIG, G.; MAKHOUL, J.; STOLCKE, A. Introduction to the Special Section on Rich Transcription. **IEEE Transactions on Audio, Speech, and Language Processing**, [S.l.], v.1490-1491, n.4, p.353–355, 2006.