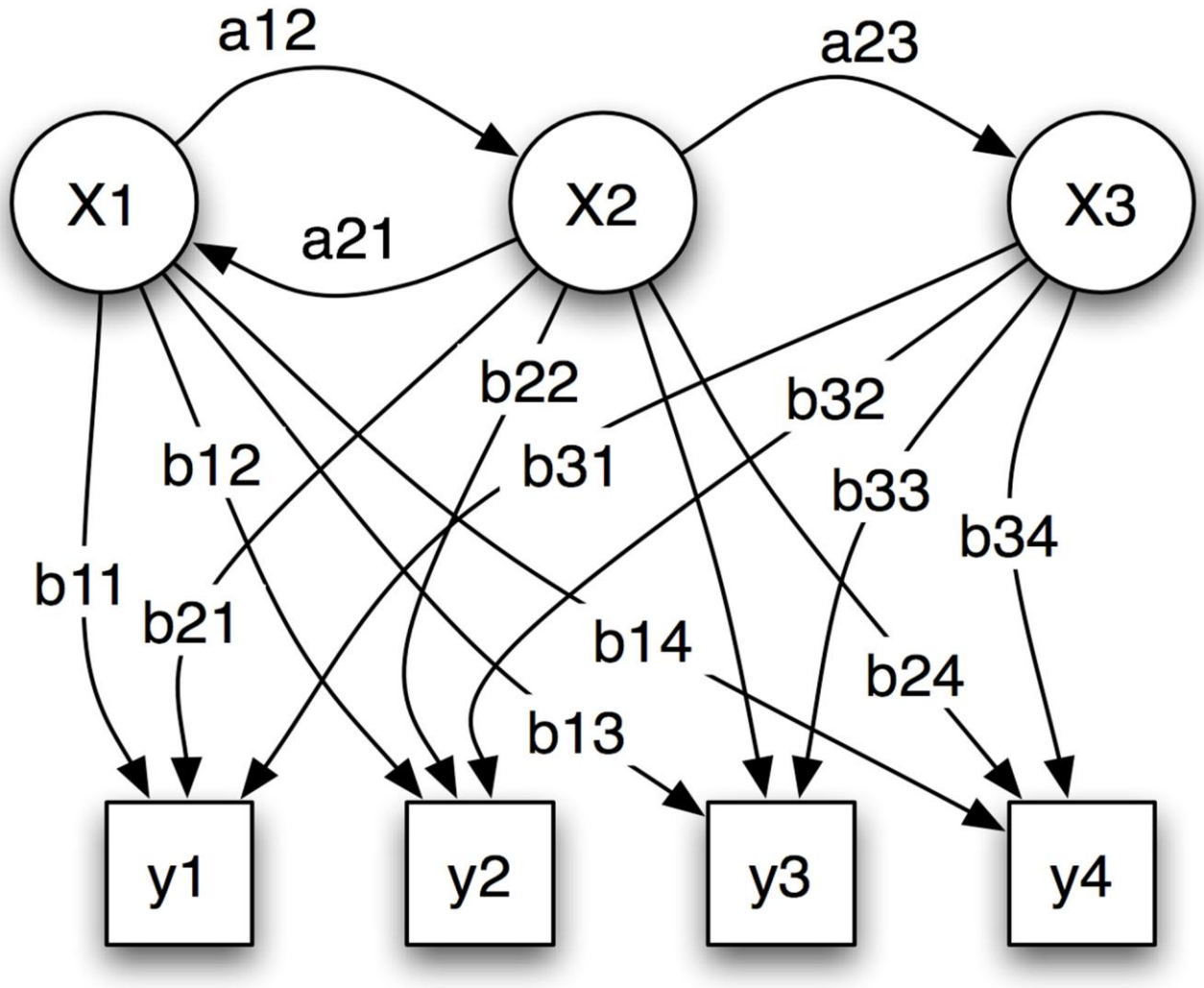

Ângela Daniela Carneiro Teixeira

DETEÇÃO E CORREÇÃO DE DISFLUÊNCIAS EM CRIANÇAS

Setembro de 2012

UNIVERSIDADE DE COIMBRA

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

MESTRADO INTEGRADO EM ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES

DETEÇÃO E CORREÇÃO DE DISFLUÊNCIAS EM CRIANÇAS

Ângela Daniela Carneiro Teixeira

Júri

Presidente: Professora Doutora Teresa Martinez dos Santos Gomes

Orientador: Professor Doutor Fernando Santos Perdigão

Vogal: Professor Doutor Marco Alexandre Cravo Gomes

Setembro de 2012

“Begin at the beginning,' the King said, very gravely, 'and go on till you come to the end: then stop.”

— Lewis Carrol, ‘Alice’s Adventures in Wonderland’

Agradecimentos

"I almost wish I hadn't gone down the rabbit-hole--and yet--and yet--..."*

Em primeiro lugar, gostaria de deixar o meu mais profundo agradecimento ao Professor Doutor Fernando Santos Perdigão por todo o apoio ao longo destes meses. Todos os conselhos e conhecimento que me transmitiu foram decisivos para a execução deste trabalho. Obrigada por nunca me ter deixado desamparada!

Gostaria também de agradecer aos colegas de laboratório, Dirce, Jorge, Arlindo e Sara, pelo companheirismo e apoio. Agradeço, em particular, à Carla Lopes pela dedicação com que me aconselhou e co-orientou.

Não posso também deixar de agradecer aos meus colegas e amigos de curso. Obrigada por todos os momentos que comigo partilharam, dentro e fora destas paredes. Num futuro próximo, comeremos mimos doces todos juntos mais uma vez!

Também aos meus amigos gostaria de deixar umas palavras de apreço. Vocês são o meu Norte e, como uma vez alguém me escreveu: 'Posso viver sem vocês, mas quando estão longe torna-se difícil respirar…'. Obrigada por me terem deixado entrar como eu vos deixei entrar também!

Resta-me agradecer à minha família: ao meu pai por ser uma pedra de força e estabilidade; à minha mãe por me ralhar e exigir de mim tudo aquilo que sabe de que eu sou capaz e por ser o meu coração; ao meu irmão por ser tão simplesmente ele, brincalhão, chato e sempre presente para mim; também ao mais recente membro da família, o meu Sleepy, a bola de pelo mais parva e carente, nos momentos mais inconvenientes.

"I can't go back to yesterday because I was a different person then."*

Obrigada a todos por me terem feito 'diferente'.

* Lewis Carrol , 'Alice's Adventures in Wonderland'

Resumo

A fala é o meio de comunicação por excelência, sendo o seu processo de aprendizagem um dos mais complexos que se conhece. No entanto, as crianças são extremamente eficientes neste processo. Ao longo do seu desenvolvimento, a criança desenvolve uma série de mecanismos mentais (ao nível da percepção e ao nível do controlo motor) que lhe permitem a aprendizagem e aperfeiçoamento da fala. No início do processo de aprendizagem, existem sempre alguns sons cuja replicação é mais difícil, sendo expectável que a criança apenas adquira o domínio de alguns fones mais complexos numa fase mais avançada do seu desenvolvimento.

Com a evolução das tecnologias desenvolvidas na área do reconhecimento automático de fala houve um investimento da sua aplicação a áreas como as da aprendizagem da língua ou da terapia da fala.

Um trabalho realizado anteriormente, que propunha o desenvolvimento de um sistema de auxílio à terapia da fala para detecção de disfluências em crianças em idade pré-escolar, serviu de ponto de partida para o presente estudo. Foi objectivo deste estudo implementar o módulo desse sistema correspondente à identificação e correcção de disfluências para sua posterior inserção no sistema completo.

Em termos de processamento do sinal de fala, o sistema que se baseia nos Modelos de Markov Não Observáveis (HMM – *Hidden Markov Models*) é um dos mais utilizados, tendo sido a escolha para o desenvolvimento de modelos de reconhecimento de fala para crianças a aplicar neste estudo.

No âmbito deste estudo procedeu-se ao tratamento da base de dados recolhida no estudo referido, para criar os modelos de reconhecimento de fala necessários para solucionar o problema principal proposto, o da identificação de dificuldades de pronunciação. Os modelos foram criados com recurso às ferramentas facultadas pelo HTK (*Hidden Markov Model Toolkit*), produzindo resultados de reconhecimento satisfatórios. Procedeu-se então ao desenvolvimento de um método que permitisse a detecção de disfluências em palavras proferidas por crianças, usando para isso os valores de verosimilhança calculados no processo de reconhecimento da locução a analisar.

Palavras-chave: Fala, reconhecimento, crianças, modelos, verosimilhança, HMM.

Abstract

Speech is the prime mean of communication, with its learning process being one of the most complex and yet, one where humans excel at. Throughout its growth, the child develops a series of mental mechanisms (either at a perception level or at a motor control level) which allow the learning and improvement of speech. At the beginning of the learning process, there are always some sounds whose replication is more difficult, it is expected that the child will only acquire the domain of some of the more complex phones at a later stage of its development.

As the technologies in the area of automatic speech recognition evolved, there was an increase in the tendency to apply it to areas such as language learning or speech therapy.

A previous work, which proposed a Computer Aided Speech Therapy system directed toward the detection of disfluencies for children in pre-school age, served as the starting point for this study. The development of this system's module that corresponded to the identification and correction of disfluencies, for subsequent insertion into the complete system, was the main goal of this thesis.

In terms of speech signal processing, the system based on Hidden Markov Models (HMM - Hidden Markov Models) is one of the most widely used and was chosen in this study for the development of models of speech recognition for children.

For the purpose of this study, the processing of the database collected in the above mentioned work was carried out. This necessary for creating the speech recognition models needed to implement this study's main goal of identifying pronunciation problems. The models were created using the tools provided by HTK (Hidden Markov Model Toolkit), having produced satisfactory recognition results. A method that allowed the detection of disfluencies on words spoken by children was also developed, using for this the likelihood values calculated in the recognition of the utterance to be tested.

Keywords: Speech, recognition, children, likelihood, HMM.

Índice

Lista de Acrónimos e Abreviações

ASR	Automatic Speech Recognition
CALL	Computer-Assisted Language Learning
DET	Detection Error Tradeoff
FSM	Finite State Machine
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
LPC	Linear Predictive Coding
MAPLR	Maximum A Posteriori Linear Regression
MFCC	Mel-Frequency Cepstrum Coeficients
MLF	Master Label File
MMF	Master Macro File
ROC	Receiver Operating Characteristic
SAMPA	Speech Assessment Methods Phonetic Alphabet
SLF	Standard Lattice Format
VTLN	Vocal Tract Length Normalisation
WER	Word Error Rate

Lista de Figuras

1. Introdução

1.1. Motivação

A fala e o seu processo de aprendizagem são um dos sistemas mais complexos que se conhece. No entanto, o ser humano fá-lo de modo inato, com extrema rapidez e eficiência. A aprendizagem de uma língua é um processo iterativo baseado em imitação, como é norma no decurso de desenvolvimento da criança. Quando esta começa a falar, surgem dificuldades em pronunciar alguns sons e palavras. Com o tempo, a criança desenvolve capacidade de controlo dos meios intervenientes na produção da fala, desenvolvendo aptidões de pronúncia e articulação. Porém, nem sempre estas dificuldades são ultrapassadas em tempos considerados normais. É nestes casos que surge a necessidade de intervenção terapêutica, de modo a impedir ou minimizar problemas na comunicação que daí advenham. Este problema torna-se particularmente pertinente quando se observam as elevadas taxas de insucesso escolar que podem ser reflexo de problemas como o acima descrito.

Desde o aparecimento dos computadores e sistemas informatizados surgiu uma tendência de aplicar as suas mais-valias aos aspectos mais vulgares da vida. A fala, sendo o meio de comunicação por excelência, não foi excepção, tendo sido motivo do desenvolvimento de inúmeros estudos e aplicações. Os sistemas de reconhecimento automático de fala (ASR - *Automatic Speech Recognition*) são talvez o expoente máximo do investimento feito neste campo. A aplicação de todo este conhecimento e tecnologia ao processo de ensino da língua surgiu de forma natural. O uso de computadores aplicado ao ensino/aprendizagem de línguas data da década de 60 sob o acrónimo CALL (*Computer Assisted Language Learning*) e faz uso de inúmeras aplicações ICT (*Information and Communication Technologies*).

Em 2002 surgiu um projecto de terapia da fala financiado pela União Europeia [1] que motivou o desenvolvimento de diversas ferramentas para auxiliar os terapeutas, para avaliação da produção de fala, entre outras.

Os sistemas de auxílio à terapia da fala podem ser segmentados em grupos, dependendo do público-alvo. Releva-se de entre eles o tratamento de problemas de dislalia[1]. Muitos destes sistemas usam tecnologias ASR na sua implementação.

1.2. Objectivos

O principal objectivo do presente trabalho foi o da criação de um sistema que possibilitasse diagnóstico de disfluências de pronunciação, dificuldade articulatória, ou mesmo dislexia em crianças em idade pré-escolar, para posterior correcção ou terapia da fala, com recurso ao reconhecimento automático de fala.

Foi ainda objectivo deste estudo criar modelos de reconhecimento de fones adaptados à faixa etária em questão, necessários para o desenvolvimento do supracitado sistema.

1.3. Organização dos Conteúdos

No capítulo 2 deste estudo apresentam-se algumas considerações sobre o contexto actual do uso de computadores adaptado à terapia de fala e dos sistemas de reconhecimento automático de fala. São ainda apresentadas algumas particularidades da fala de crianças e expõe-se uma visão geral sobre o sistema a desenvolver.

O capítulo 3 contém uma breve exposição sobre uma das técnicas mais comummente utilizadas em reconhecimento automático de fala, os Modelos de Markov Não-Observáveis (HMM – *Hidden Markov Models*), e sobre uma técnica de parametrização espectral.

No capítulo 4 apresenta-se o desenvolvimento do processo de criação dos modelos de fones, incluindo o tratamento da base de dados e a parametrização dos ficheiros de áudio.

O capítulo 5 inclui a descrição do sistema de verificação e identificação de disfluências desenvolvido, bem como a proposta de uma solução para o problema de classificação binária inerente ao problema de identificação de disfluências.

No último capítulo são apresentadas as conclusões, a análise dos resultados obtidos e propostas sobre a direcção trabalhos futuros de continuidade.

[1] Dificuldades na fala de origem não neurológica associadas aos órgãos externos de produção de fala [26].

2. O Ensino Assistido da Língua

2.1. Tecnologia Actual

O uso de computadores aplicado ao ensino/aprendizagem de línguas é conhecido sob o acrónimo CALL. A fase em que a tecnologia das aplicações CALL se encontra varia de acordo com a perspectiva de quem faz esta análise. Por exemplo, enquanto Warschauer defende que, de 2000 em diante, se entrou na fase 'Integrativa'[2] [2], já Bax [3] considera que a análise anterior possui falhas e algumas contradições e propõe uma nova visão onde considera que a presente fase é a 'Open CALL'[3]. Este defende ainda que só deveria ser considerada a fase 'Integrativa' quando as aplicações CALL forem de uso tão simples e comum como os mais tradicionais e correntes métodos de ensino. A tabela seguinte resume a opinião de Warschauer sobre as fases das aplicações CALL. A Tabela 2 apresenta a visão de Bax sobre o mesmo assunto.

Tabela 1 – As fases das aplicações CALL segundo Warschauer [2].

Stage	1970s-1980s: Structural CALL	1980s-1990s: Communicative CALL	21st Century: Integrative CALL
Technology	Mainframe	PCs	Multimedia and Internet
English-Teaching Paradigm	Grammar-Translation & Audio-Lingual	Communicate Language Teaching	Content-Based, ESP/EAP
View of Language	Structural (a formal structural system)	Cognitive (a mentally-constructed system)	Socio-cognitive (developed in social interaction)
Principal Use of Computers	Drill and Practice	Communicative Exercises	Authentic Discourse
Principal Objective	Accuracy	Fluency	Agency

(Based on Kern & Warschauer, 2000; Warschauer, 1996; Warschauer, 2000a)

.

[2] Fase 'Integrative' – baseada na alargada dispersão de meios devido aos grandes desenvolvimentos dos computadores e da Internet.

[3] 'Open CALL' – aberto em todas as dimensões desde o *feedback*, tipos de software, papel do professor, particularmente quando comparado com a fase 'Restricted CALL'.

Tabela 2 As fases CALL de acordo com Bax [3].

Table 2
Restricted, Open and Integrated CALL: an outline

Content	Type of task	Type of student activity	Type of feedback	Teacher roles	Teacher attitudes	Position in curriculum	Position in lesson	Physical position of computer
Restricted CALL								
Language system	Closed drills Quizzes	Text reconstruction Answering closed questions Minimal interaction with other students	Correct/incorrect	Monitor	Exaggerated fear and/or awe	Not integrated into syllabus—optional extra Technology precedes syllabus and learner needs	Whole CALL lesson	Separate computer lab
Open CALL								
System and skills	Simulations Games CMC	Interacting with the computer Occasional interaction with other students	Focus of linguistic skills development Open, flexible	Monitor/facilitator	Exaggerated fear and/or awe	Toy Not integrated into syllabus—optional extra Technology precedes syllabus and learner needs	Whole CALL lesson	Separate lab—perhaps devoted to languages
Integrated CALL								
Integrated language skills work Mixed skills and system	CMC WP e-mail Any, as appropriate to the immediate needs	Frequent interaction with other students Some interaction with computer through the lesson	Interpreting, evaluating, commenting, stimulating thought	Facilitator Manager	Normal part of teaching—normalised	Tool for learning Normalised integrated into syllabus, adapted to learners' needs *Analysis of needs and context precedes decisions about technology*	Smaller part of every lesson	In every classroom, on every desk, in every bag

O desenvolvimento de aplicações CALL é também assegurado por associações profissionais, a maior parte das quais sem fins lucrativos, que se dedicam a promulgar e discutir assuntos relacionados com a investigação e desenvolvimento das tecnologias de ensino de línguas. De entre estas podem referir-se algumas como a APACALL (The Asia-Pacific Association for CALL), a CALICO (The Computer Assisted Language Instruction Consortium), a EUROCALL (European Association for Computer Assisted Language Learning), ou a JATCALL (The Japan Association for Language Teaching - CALL).

De entre as aplicações CALL, têm interesse para o presente estudo aquelas que usam o ensino da língua falada. Neste contexto as aplicações CALL consideram-se divididas em dois grupos: aquelas que se assentam maioritariamente na tecnologia do reconhecimento de fala e as que usam apenas a síntese de fala.

No presente estudo, as tecnologias ASR (Automatic Speech Recognition) são de extrema importância uma vez que se pretende perceber se as crianças têm ou não problemas de dicção. A confiabilidade e a eficiência com que a aplicação a desenvolver é capaz de reconhecer correctamente a fala, dita o sucesso da aplicação desenvolvida.

Actualmente, existem vários pacotes de *software* comerciais destinados ao ensino de línguas,

No que diz respeito aos produtos CALL presentes no mercado, o Rosetta Stone [4] é provavelmente um dos mais conhecidos e bem-sucedidos, contendo aplicações que fazem uso de ASR (proprietário). Abrange uma ampla gama de línguas com uma vasta selecção de exercícios e tem tido uma boa aceitação por parte dos utilizadores.

A empresa LANGMaster distribui cursos de línguas *online* gratuitamente em diversos idiomas, sendo que alguns dos exercícios utilizam a tecnologia de reconhecimento de fala ViaVoice [5] da IBM[5]. Oferece centenas de horas de aprendizagem gratuitas em 75 idiomas, incluindo Português, embora apenas possua aplicações que fazem recurso a sistemas ASR para inglês. Babbel [6] é outro *software* de ensino de línguas que pode ser usado *online* e faz uso de ASR [7], necessitando o utilizador apenas do *browser* de Internet e da última versão do Adobe Flash Player. É multiplataforma e suporta vários idiomas, incluindo Português do Brasil (a norma PT-BR).

[5] - O sistema de reconhecimento desenvolvido pela IBM foi absorvido pela Nuance, já não existindo de forma independente.

Embora os programas CALL proliferem no mercado, os algoritmos de reconhecimento de fala atingem o seu auge em termos de eficiência e confiabilidade em aplicações para dicção de texto e aplicações de controlo do computador por fala.

O produto 'Dragon NaturallySpeaking', agora da Nuance [8] encontra-se, de longe, no topo [9] da lista no que diz respeito a eficiência e desempenho em reconhecimento da fala. A Nuance adquiriu o sistema ViaVoice da IBM, tornando-se líder de mercado. Também a companhia LumenVox [10] fornece soluções comerciais para aplicações de reconhecimento de fala.

A Microsoft também fez apostas nesta área, possuindo um sistema de reconhecimento de fala incorporado nos seus sistemas operativos desde o Windows Vista, tendo visto o seu desempenho e recepção melhorar com as optimizações incluídas no Windows 7 [11] [12].

Os produtos comerciais mencionados e a actual tecnologia ASR em geral, têm respostas bastante boas para os objectivos a que se propõem, embora os seus resultados com utilizadores não nativos (diferenças na pronúncia) ou com crianças, sofram uma considerável quebra.

Desde que surgiram estudos apontando o facto do fraco desempenho dos sistemas ASR com crianças, houve um incremento na atenção e interesse nessa área, tendo aparecido posteriormente muitos estudos sobre este assunto.

2.2. Considerações Sobre a Fala de Crianças

Uma vez que as tecnologias de reconhecimento de fala apresentam limitações quando aplicadas à fala de crianças, tem interesse referir alguns aspectos fundamentais sobre as características da fala das crianças. De salientar dois desses aspectos: o tom e os formantes. O tom da fala é a frequência de vibração das cordas vocais que é especialmente alta nas crianças. Os formantes são os picos espectrais associados às ressonâncias do tracto vocal [13].

A análise LPC (Linear Predictive Coding) é usada frequentemente para representar um sinal na sua envolvente espectral, e é uma das ferramentas mais eficientes na análise de voz. A sua aplicação a sinais de fala permite uma fácil observação dos formantes, como é de resto perceptível na Figura 2. 1, representativa de uma vogal 'i'. O conhecimento das posições espectrais dos formantes F1 e F2 (os formantes com as frequências mais baixas), possibilita a distinção das vogais, determinando as suas características como a altura (posição vertical da língua) – vogal aberta ou fechada - e a posteridade (posição horizontal da língua) – anterior ou posterior [14].

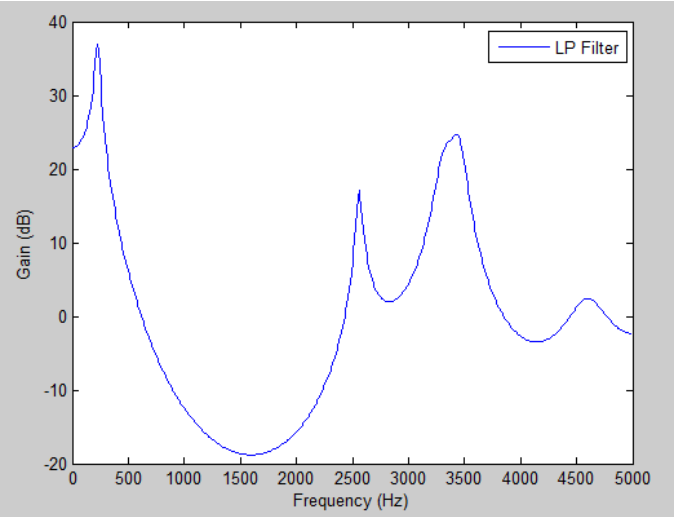

Figura 2. 1 – Gráfico LPC de uma trama da vogal 'i'

No caso de crianças, os formantes possuem valores bastante mais elevados que no caso de adultos, aparecendo os picos na frequência que representam os formantes com frequências maiores. Este facto é devido ao reduzido tamanho do seu tracto vocal. Além disso a frequência de tom é também mais elevada que no caso de adultos, o que causa um aumento no espaçamento entre harmónicos [15] [16] em sons vozeados. Enquanto num adulto os três primeiros formantes se encontram geralmente entre os 300 Hz e os 3.2 kHz, para uma criança apenas dois podem ser encontrados nesta gama [17]. Estas particularidades, entre outras, provocam uma deterioração nos resultados dos sistemas de reconhecimento automático de fala.

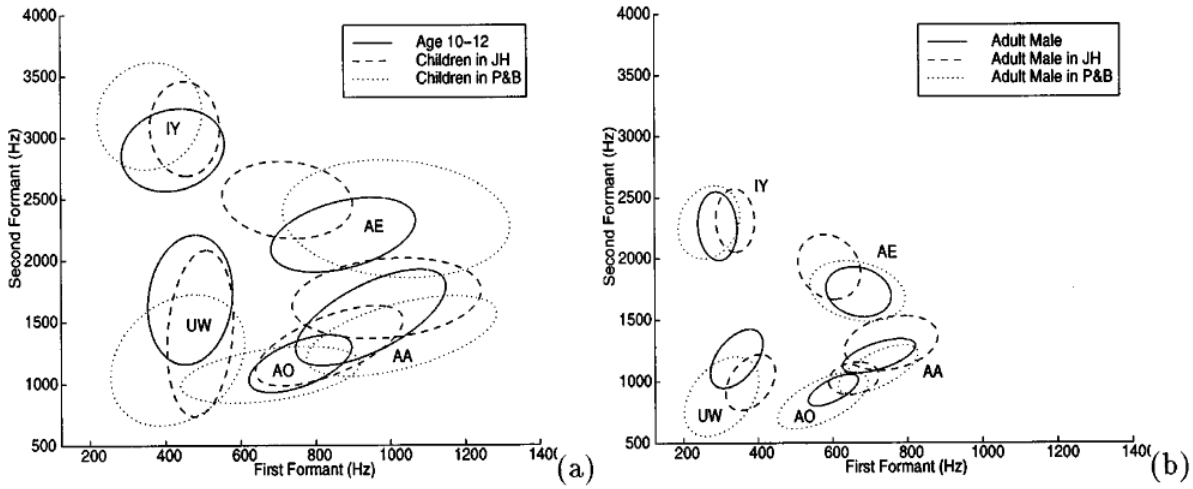

Figura 2. 2 – Posicionamento das vogais em função de F1-F2 para crianças (a) e adultos (b) [18]

Como é visível na figura anterior, há um considerável aumento da variação da frequência dos formantes com o decréscimo da idade. Também a Figura 2. 3 apresenta a variação espectral dos formantes com a variação da idade. É possível observar um decréscimo nos valores de F1 e de F2.

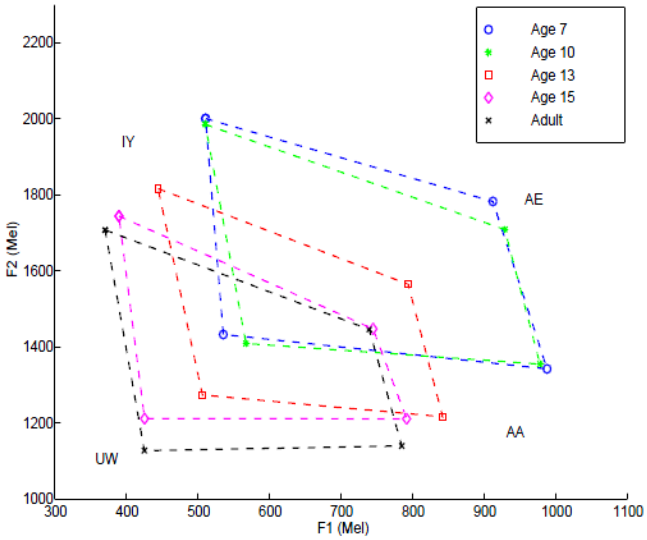

Figura 2. 3 – Variação da frequência de F1 e F2 com a idade [15].

Adicionalmente, crianças com idades inferiores a 10 anos apresentam uma maior variação da duração de vogais comparativamente aos valores normais em adultos além de uma maior variação espectral [19].

Além destes aspectos, a fala das crianças apresenta fortes variações provocadas pelas alterações de crescimento e desenvolvimento, que são atribuídas a diferenças anatómicas e morfológicas na geometria no tracto vocal. Adicionalmente, as crianças têm, de um modo geral, menor controlo dos articuladores e variações de prosódia muito acentuadas. Estudos foram realizados relacionando as frequências fundamentais e dos formantes com a idade [20], impulsionando a aplicação de algumas técnicas para o desenvolvimento de aplicações mais robustas.

Estas variações apresentam-se como dificuldades na obtenção de bons resultados quando se pretende fazer o reconhecimento automático da fala das crianças. Algumas técnicas podem ser aplicadas para minimizar estes efeitos como é o caso do VTLN (*Vocal Tract Length Normalization*) ou 'frequency warping' em que se faz uma distorção em frequência dos parâmetros espectrais de forma a apresentarem valores de referência. A utilização de modelos para normalização da frequência dos formantes em vogais de crianças para os níveis de adultos foi considerada em alguns estudos [21], [22], mostrando resultados aquém dos obtidos com fala

de adultos, particularmente para crianças mais novas. Mostram ainda diferenças nos valores dos formantes no que diz respeito ao género, durante o período de desenvolvimento.

Outra destas técnicas de normalização é conhecida como MAPLR (*Maximum A Posteriori Linear Regression*), os resultados produzidos, apesar de revelarem melhorias notórias no desempenho, situam-se aquém dos desejáveis [19] [23] [24] [17]. O estudo [25] representa um bom resumo dos avanços conseguidos nesta área.

2.3. Proposta do Sistema a Desenvolver

A proposta de criação de um 'Sistema para Terapia da Fala Auxiliada por Computador' [26] serviu como ponto de partida da presente dissertação. O sistema pretende auxiliar (não substituindo) o terapeuta de fala, na identificação de dificuldades de articulação de fonemas, fazendo para isso uso da tecnologia do reconhecimento automático de fonemas num ambiente lúdico. O esquema do sistema proposto pode ser observado na figura a seguir apresentada.

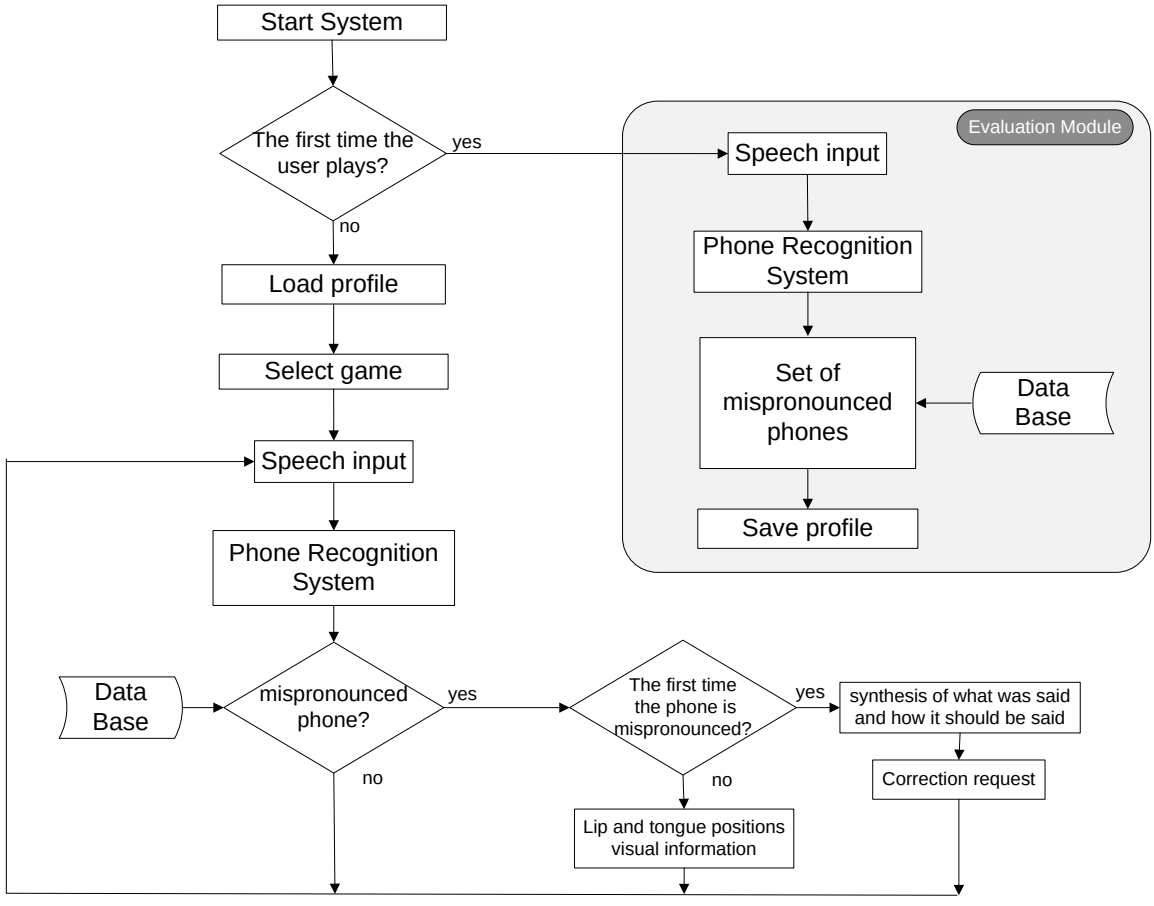

Figura 2. 4– Diagrama de blocos do sistema proposto [26].

Este sistema propõe vários módulos que se articulam de modo a poder analisar a fala de uma criança, identificando os seus problemas de dicção, e fornecer auxílio descritivo (informação visual) para a correcção das disfluências detectadas.

No presente estudo foi implementado o módulo de reconhecimento de fones para fala de crianças (através de modelos gerados e treinados para o efeito) e um módulo de verificação da correcta pronunciação de palavras.

A aplicação de um sistema de reconhecimento de fones para fala de crianças revela-se assim de inegável importância na elaboração e desenvolvimento do sistema. Para a correcta identificação de dificuldades de articulação das crianças em idade pré-escolar, é essencial que o reconhecimento do que a criança disse seja o mais preciso possível e que o sistema de verificação detecte correctamente os acertos e erros de articulação.

Para a criação do referido sistema foi necessário criar modelos para o reconhecimento de fones, especificamente para a faixa etária em questão. Estes modelos foram gerados a partir de uma base de dados de fala de crianças recolhida no âmbito de um projecto anterior [26]. A descrição desta base de dados será detalhada em pontos posteriores deste estudo (secções 4.1 e 4.1.1).

Torna-se, neste contexto, importante definir o conceito de fone e fonema. No contexto da fonética, um fone é a unidade mínima acústica do sistema fonético, que corresponde à realização física ou real de um fonema. São os sons que são distinguíveis pelo seu contraste dentro de uma palavra. Um fonema representa então uma abstracção do fone, sendo considerado a unidade básica de uma língua. [27] [28].

3. Técnicas de Análise e Reconhecimento da Fala

O problema do reconhecimento automático de fala não se encontra ainda completamente resolvido, particularmente no que respeita à fala contínua (por oposição ao reconhecimento de palavras isoladas). Porém, como foi já referido no capítulo anterior, apresenta bons resultados (nomeadamente para fala de adultos). Neste capítulo faz-se uma descrição sumária sobre uma das técnicas mais utilizadas em reconhecimento automático de fala, a modelação com os modelos de Markov não observáveis, bem como a técnica de parametrização espectral dos sinais de fala.

3.1. Coeficientes Cepstrais em Escala Mel

O *cepstrum* em escala mel é um esquema de representação de sinais largamente utilizado na análise de sinais de fala. Tem-se vindo a tornar uma escolha bastante popular para extracção de características devido à sua capacidade de caracterizar eficientemente o sinal de fala [29].

A definição original de *'Cepstrum'* foi proposta em 1963 [30] e consiste na transformada inversa de Fourier do logaritmo do espectro estimado de um sinal, sendo os seus coeficientes calculados através da seguinte fórmula:

$$c_x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \log\left|X\left(e^{j\omega}\right)\right| e^{j\omega n} d\omega \qquad (1)$$

Os coeficientes cepstrais em escala mel[6], conhecidos pela sigla MFCC (*Mel-Frequency Cepstrum Coefficients*), diferem dos normais coeficientes de *cepstrum* em dois aspectos essenciais: o uso de uma descrição espectral baseada num banco de filtros e no facto de as bandas de frequência destes filtros se encontrarem igualmente espaçadas na escala mel. A expressão (2), proposta por Douglas O'Shaughnessy [31], estabelece a conversão de frequência de uma escala linear para a escala mel. Este mapeamento aproxima-se da resposta do sistema auditivo humano que tem uma relação não linear entre a percepção do tom e a frequência, sendo esta relação aproximadamente linear até cerca de 700Hz e logarítmica para frequências superiores [29].

[6] Escala perceptual de tons onde os ouvintes definem os tons espaçados igualmente.

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \quad (2)$$

Os filtros apresentam uma forma triangular cobrindo toda a gama de frequências de interesse. De notar ainda o facto de as larguras de banda dos filtros serem crescentes com o aumento da frequência segundo a escala mel, adaptando-se, assim, à diminuição de resolução de frequência do ouvido humano.

Os coeficientes MFCC são, desde há longa data, os parâmetros escolhidos em sistemas de reconhecimento automático de fala, uma vez que se traduzem numa descrição compacta do sinal de fala com boas características de redução de ruído. Devido a estas características, foram então os parâmetros escolhidos para a análise das locuções de fala das crianças.

3.2. Hidden Markov Models

Os modelos HMM (Hidden Markov Models) são modelos estatísticos apresentados inicialmente por Baum [32] e seus colaboradores em 1966, cuja ampla e reconhecida aplicação em reconhecimento automático da fala foi em muito impulsionada por Rabiner nos seus estudos [33] [34].

O sinal de fala é um sinal não estacionário que apresenta variações na sua composição espectral. Os modelos de Markov podem ser usados para descrever esta variação temporal de propriedades do sinal, assumindo um conjunto finito de configurações espectrais ou estados. Um HMM é um processo que descreve um sinal como uma cadeia de N estados (S_1, S_2, …, S_N), alterando o seu estado a intervalos regulares de tempo, podendo mudar de estado ou permanecer nele próprio. A probabilidade de transição de um estado para cada um dos outros possíveis estados encontra-se definida numa matriz de distribuição de probabilidade – Matriz de Transição (também conhecida como Matriz de Markov - M) – com elementos a_{ij} (a probabilidade de transitar do estado *i* para o estado *j*). A transição para um novo estado fica assim, apenas dependente do estado anterior. A figura seguinte representa um modelo HMM simples com as possíveis transições e probabilidades associadas, sequências observadas ($o_1, o_2, \ldots, o_6$) e suas probabilidades de observação ($b_j(o_t)$).

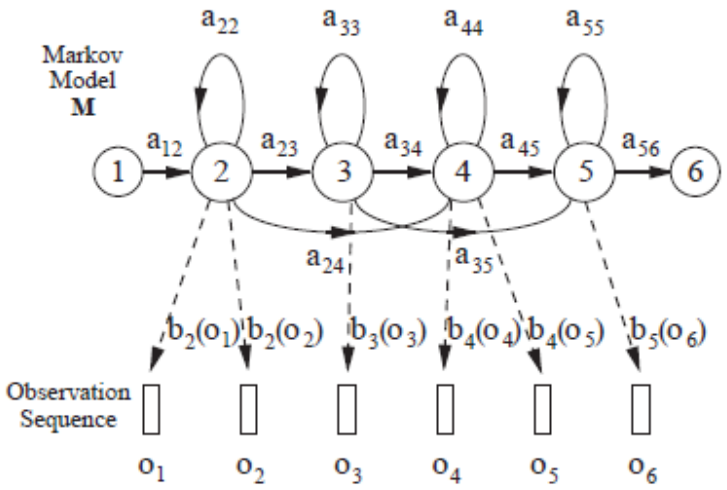

Figura 3. 1 – Exemplo de um modelo HMM simples com topologia esquerda-direita [35].

Assim, cada estado pode gerar uma sequência de observações, de entre um conjunto, de acordo com uma determinada distribuição que é associada a cada estado do modelo. É portanto possível obter a mesma sequência de observações, com probabilidades diferentes, através de diferentes sequências de estados. No caso dos Modelos Não Observáveis (*Hidden Markov Models*), a sequência de estados do sistema não é conhecida, apenas a sequência de observações gerada. Define-se então $b_j(x_t)$ como essa distribuição de probabilidade associada a cada estado: a distribuição condicional de observar o vector de características do sinal (observação x_t) no instante t para o estado j. Para sistemas modelados por distribuições contínuas, a função densidade de probabilidade contínua é composta por uma mistura pesada de componentes Gaussianas (GMM – *Gaussian Mixture Model*).

Um HMM é, portanto, um duplo processo estocástico com uma cadeia de Markov e funções densidade de probabilidade por estado, que pode ser representada pelas equações (3) e (4), onde:

- Q representa a dimensão do vector de observação **x**;
- M - é o número de componentes Gaussianas;
- $\boldsymbol{\mu}_{jm}$ - é o vector de médias da componente m do GMM do estado j;
- $\boldsymbol{\Sigma}_{jm}$ - é a matriz de covariância da componente m do GMM do estado j;
- c_{jm} - é o peso de cada componente da mistura;
- $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{jm}; \boldsymbol{\Sigma}_{jm})$ - é a função de densidade normal (Gaussiana).

$$b_j(\mathbf{x}) = \sum_{m=1}^{M} c_{jm}\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{jm},\boldsymbol{\Sigma}_{jm}), \qquad \sum_{m=1}^{M} c_{jm} = 1 \tag{3}$$

$$\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{jm},\boldsymbol{\Sigma}_{jm}) = \frac{1}{\sqrt{(2\pi^{Q})|\boldsymbol{\Sigma}_{jm}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_{jm})^{T}\boldsymbol{\Sigma}_{jm}^{-1}(\mathbf{x}-\boldsymbol{\mu}_{jm})\right) \tag{4}$$

Para efeitos de aplicação em sistemas ASR, a topologia usada é a esquerda-direita, onde apenas são autorizadas transições para o mesmo estado ou para o estado subsequente, o que é facilmente perceptível quando se analisa o processo de produção da fala como uma sequência de fones.

Considera-se ainda que a produção da sequência de observações **x** resulta de um HMM, onde apenas é observável a sequência emitida por cada estado e não a sequência de estados inerente ao modelo. Torna-se então possível estimar iterativamente os parâmetros do modelo usando um número considerável de sequências de observação e, por conseguinte, estimar a mais provável sequência de estados e a probabilidade de esta ter sido gerada por um dado modelo.

A figura seguinte apresenta um modelo HMM de topologia esquerda-direita para representar um fonema, com GMMs e observações associadas aos estados.

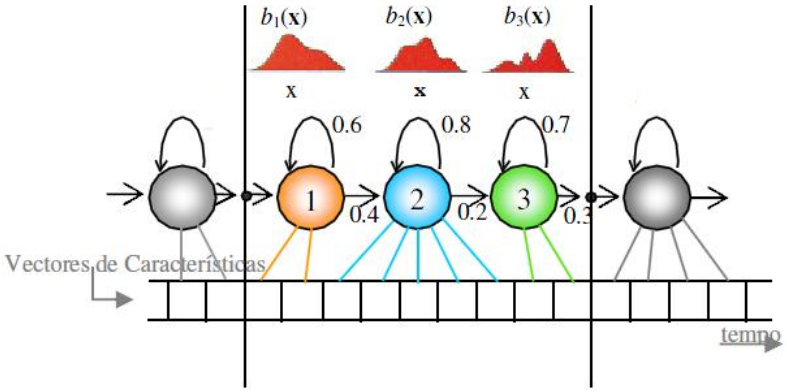

Figura 3. 2 – Exemplo de um modelo HMM com topologia esquerda-direita e com funções densidade de probabilidade compostas por GMMs [26].

Neste contexto surge o algoritmo de Viterbi [36] [37], de relevante importância neste estudo pois é usado na ferramenta `HVite`, facultada pelo HTK (*Hidden Markov Model Toolkit*)[7], para efeitos de alinhamento e reconhecimento. Este algoritmo de programação dinâmica calcula a sequência de estados mais provável para uma dada sequência de observações (sendo essa

[7] Toolkit criado na Universidade de Cambridge para a geração e manipulação de HMM.

sequência conhecida como o 'caminho de Viterbi'), podendo então ser considerado um algoritmo de descodificação da máxima verosimilhança. O algoritmo acha o caminho mais provável através de uma *trellis*, um gráfico de estados e tramas, ou seja, calcula o caminho mais curto num conjunto de observações, como exemplifica a figura seguinte.

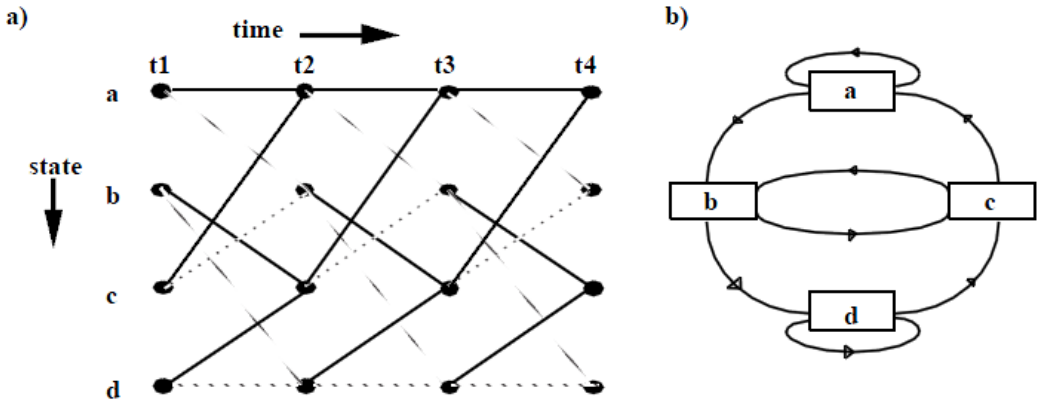

Figura 3. 3 – Exemplo de um diagrama *trellis* (a) e do HMM que o originou (b) [38].

Cada ponto na imagem anterior representa a probabilidade de um estado ser observada naquele instante e os arcos representam a probabilidade de cada transição. A probabilidade de cada caminho é então calculada através da soma das probabilidades de transições e as probabilidades de *output* ao longo do caminho.

De relevante importância é também o algoritmo Baum-Welch [39], que calcula os parâmetros de um HMM através do algoritmo *'forward-backward'*. Este algoritmo usa o critério de maximização da verosimilhança para estimar os parâmetros. Em geral, para um conjunto fixo de dados e o modelo estatístico subjacente, o método de máxima verosimilhança selecciona valores dos parâmetros do modelo, que produzem uma distribuição que fornece os dados observados com maior probabilidade (ou seja, os parâmetros que maximizam a função de probabilidade). O valor do logaritmo da verosimilhança (*log-likelihood*) é então calculado usando a função de densidade de probabilidade, como se pode observar na equação seguinte.

$$\sum_{i=1}^{n} \ln\left(f\left(x_i \mid \theta\right)\right) \tag{5}$$

4. Reconhecimento de Fala de Crianças

Como dito anteriormente, um aspecto essencial deste trabalho é a criação de modelos acústicos de fonemas, aplicados a fala de crianças, para o sistema de verificação de disfluências. Para criar estes modelos é necessária uma base de dados de fala de crianças. A secção seguinte descreve a base de dados usada e as outras secções descrevem o processo de criação de modelos.

4.1. Recolha de uma Base de Dados de Fala de Crianças

A inexistência de corpora para fala de crianças em português Europeu propulsionou a criação de uma base de dados que pudesse servir o propósito da definição de modelos para sistema como o supra-referido. Esta base de dados, usada no presente trabalho, foi recolhida em 2010, no âmbito de um estudo anterior, e está descrita com detalhe em [26]. Os dados de áudio foram recolhidos em infantários da zona centro de Portugal com um total de 111 crianças, 55 do sexo masculino e 56 feminino, com idades compreendidas entre os 5 e os 6 anos. A cada criança foi pedido que identificasse 40 palavras (representadas por imagens) escolhidas aleatoriamente de um conjunto de 200 palavras, o que resultou num total de 3726 ficheiros válidos. A frequência de amostragem usada foi de 8 kHz. A figura seguinte representa o histograma da distribuição dos diferentes fones presentes na base de dados.

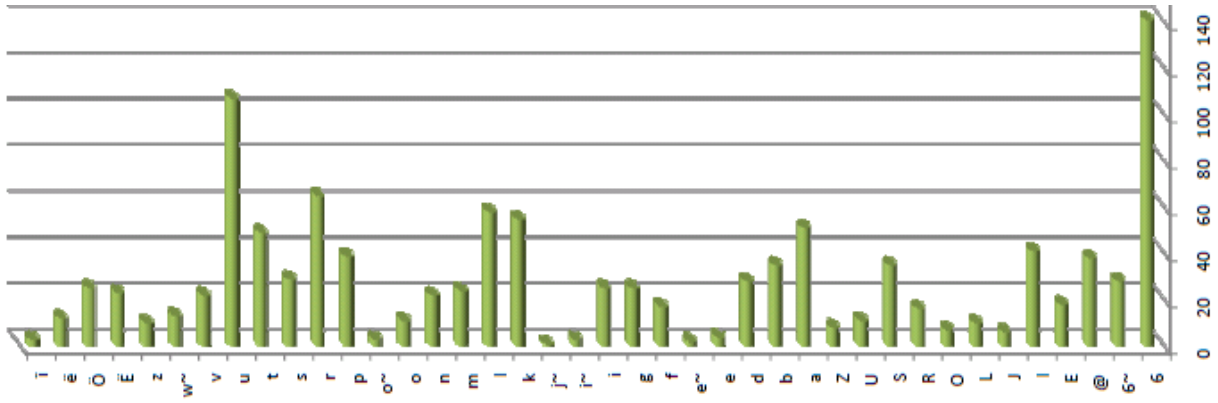

Figura 4. 1 - Ocorrência de cada fone na lista de 200 palavras.

As condições da recolha das locuções e o facto de se lidar com crianças comprometeram, de alguma forma, a qualidade e as condições acústicas das gravações, o que implicou uma anotação da base de dados com bastante detalhe. No âmbito deste projecto as etiquetas ortográficas desta base de dados foram reanalisadas com o intuito de as verificar e eliminar inconsistências.

4.1.1 Tratamento da Base de Dados

Inicialmente, todos os ficheiros de áudio (‘wav’) foram filtrados com um filtro passa-alto com frequência de corte de 100 Hz, num script de Matlab, com o intuito de lhe retirar parte dos ruídos de baixa frequência. Também, por existirem inconsistências em termos temporais e de etiquetas atribuídas, e até alguns erros ortográficos, houve necessidade de rever todos os ficheiros da base de dados. Para tal foi usado o um *software* de anotação, em ambiente Matlab (‘`Viewlabs`’, ver Figura 4. 2), que permitiu a audição, visualização e edição das etiquetas e dos seus tempos.

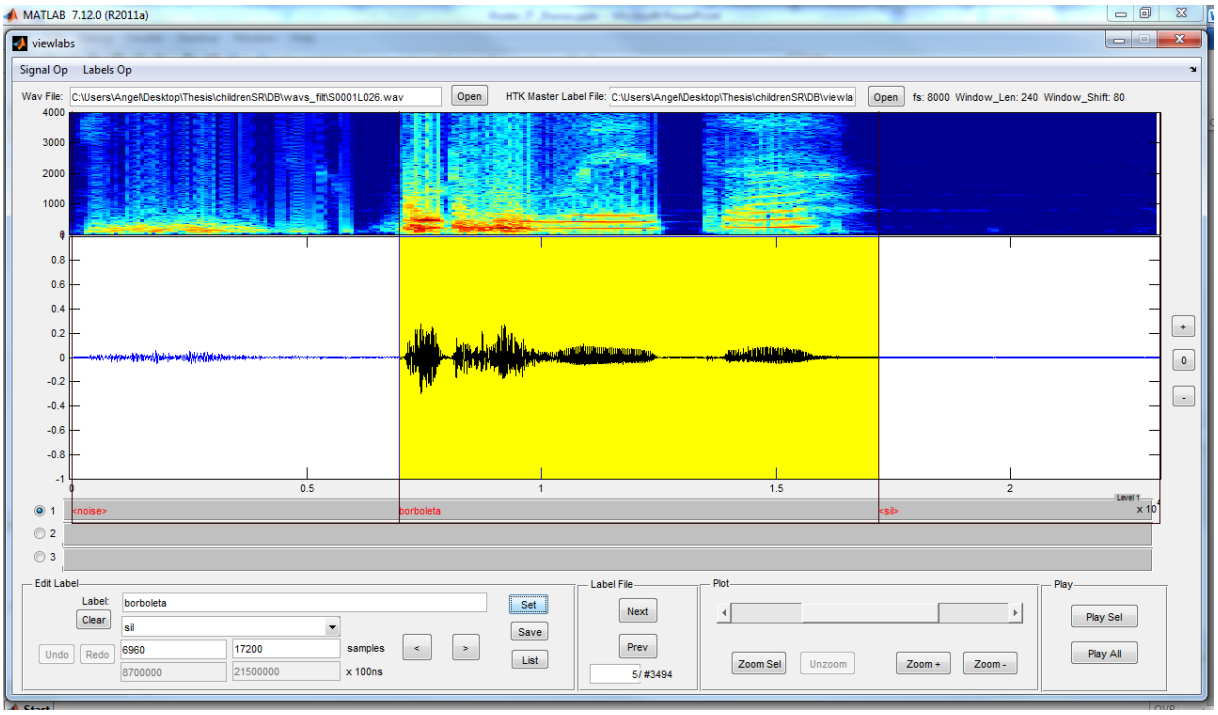

Figura 4. 2 – Viewlabs em execução com um ficheiro de áudio e as respectivas etiquetas e marcações temporais

Os ficheiros de anotação seguem um formato definido pelo *software* de criação de modelos acústicos, o HTK. Trata-se de ficheiros de texto com indicação de tempos iniciais e finais para cada etiqueta. A cada ficheiro de áudio corresponde um ficheiro de etiquetas com o mesmo nome e uma extensão diferente (‘.lab’). É possível compilar as transcrições de muitos ficheiros de áudio num único ficheiro, chamado ficheiro "mestre" (MLF – *Master Label File*). Este ficheiro possui, como se pode observar na Figura 4. 3, uma lista de transcrições ‘.lab’ (a indicação de início de ficheiro é representada pelo nome entre aspas). Cada uma das etiquetas presente nos ficheiros é representada por um tripleto (tempo inicial, tempo final e a própria etiqueta) sendo o fim de cada transcrição indicado por um ponto final.

```
#!MLF!#
"*/S0001L047.lab"
0          7600000    <sil>
7600000    17000000   cenoura
17000000   19200000   <breath>
19200000   25998750   <sil>
.
"*/S0001L060.lab"
0          9800000    <sil>
9800000    20500000   copo
20500000   26998750   <sil>
.
"*/S0001L068.lab"
0          9500000    <sil>
9500000    20300000   espada
20300000   25000000   <noise>
25000000   28998750   <sil>
.
```

Figura 4. 3 – Exemplo de um ficheiro MLF (excerto).

No âmbito do presente trabalho todos os ficheiros áudio foram atentamente ouvidos, sendo as respectivas etiquetas ortográficas verificadas. Nesta fase foram feitas correcções, nomeadamente algumas etiquetas foram redefinidas e os seus limites temporais reajustados.

A notação usada nas etiquetas foi normalizada para facilitar o seu processamento. As etiquetas são definidas de forma ortográfica com a transcrição das palavras ditas pelas crianças. Além disso, foram usadas outras etiquetas para representar eventos extralinguísticos, resultando na seguinte lista de etiquetas com a respectiva correspondência em termos de modelos de eventos.

- <breath> resp
- <garbage> noise
- <laugh> noise
- <noise> noise
- <resp> resp
- <resp_exp> resp
- <resp_insp> resp
- <sil> sil

Adicionalmente, procedeu-se à catalogação das locuções quanto à sua qualidade (condições de ruído e/ou inteligibilidade) através de um processo manual onde se adicionaram caracteres específicos no fim das transcrições de modo a poder construir as marcações de ruído. Deste

processo resultaram diferentes versões do ficheiro de etiquetas (de acordo com o nível de ruído e de inteligibilidade).

O último e mais limpo ficheiro MLF (em termos de qualidade e percepção do áudio, com o nome 'ChildCAST_clean.mlf') tem 2650 transcrições, tendo sido removidas todas as locuções/transcrições que possuíam demasiado ruído e cuja inteligibilidade era baixa [8].

A partir deste ficheiro MLF foi produzida uma lista com as ocorrências de todas as etiquetas, incluindo as palavras correctas (palavras bem pronunciadas), erros de pronunciação/articulação, bem como as palavras de ligação (artigos, conjunções e verbos, como por exemplo em "é uma bruxa") e sons/palavras de hesitação (ver *script* `'formatmlf.m'`, presente na secção de anexos 7.4).

Um dicionário com todas as palavras da lista (num total de 522 ocorrências, das quais 346 foram consideradas palavras correctas) e sua tradução no alfabeto fonético SAMPA[9], foi então gerado. O alfabeto tem 33 fones que estão listados no anexo 7.2. A transcrição em SAMPA foi inicialmente feita com o auxílio do conversor de grafemas para fonemas (G2PConverter.bat) entretanto desenvolvido no laboratório [40]. Posteriormente todas as transcrições foram manualmente verificadas e corrigidas quando necessário. A este dicionário foram adicionadas as etiquetas extralinguísticas e variações que permitem multipronunciação em algumas palavras, como demonstra o exemplo da figura seguinte.

```
caramelo          k a r a m e l u
caranguejo        k a r a~ g a Z u
caranguejo        k a r a~ g a i Z u
carrinha          k a R i J a
carro             k a R u
carrocel          k a R O s E l
```

Figura 4. 4 – Excerto do dicionário com tradução SAMPA e com exemplo de multipronunciação.

4.1.2 Parametrização da Base de Dados

[8] Este resultado foi obtido com o script de Matlab `'cleannoisemlf.m'` (ver anexo), que remove as etiquetas com as marcações de ruído, seguido pelo script `'emptynoisemlf.m'`, que remove do MLF todos as etiquetas que contêm apenas marcas acústicas (sem locuções).

[9] SAMPA (Speech Assessment Methods Phonetic Alphabet) – sistema de escrita fonético que tem como base o Alfabeto Fonético Internacional (IPA).

Procedeu-se então à fase de parametrização dos ficheiros de áudio com MFCC. Devido às características do sinal de fala típico de uma criança, cujos formantes e tom são muito superiores aos de um adulto, optou-se por um espaçamento de filtros de 143 mel, o que se traduziu na aplicação de 14 filtros, na escala mel desde 100 Hz até metade da frequência de amostragem - 4 kHz (ver Figura 4. 5 e Figura 4. 6). Este valor do espaçamento foi calculado dividindo a gama de frequências de interesse pelos 14 filtros. Abaixo de 100 Hz considera-se que as características espectrais são irrelevantes.

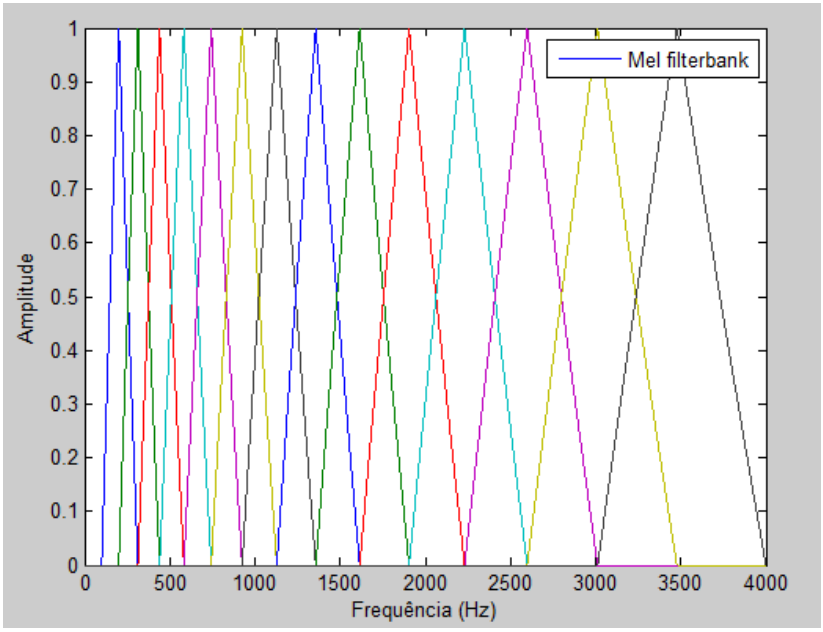

Figura 4. 5 – Representação dos 14 filtros na frequência.

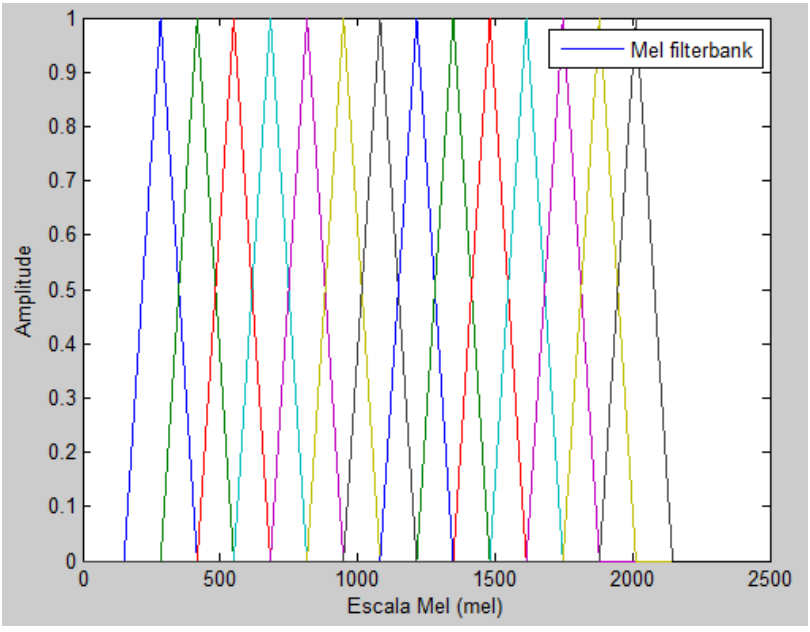

Figura 4. 6 - Representação dos 14 filtros da figura anterior em escala MEL.

Numa primeira abordagem, utilizou-se o Praat[10] [41] para calcular os MFCCs. Foram criados *scripts* para correr todos os ficheiros áudio e, para cada, recolher a informação relativa à harmonicidade[11], tom e os próprios MFCCs.

Pelo facto de a ferramenta de reconhecimento escolhida, onde se iria proceder ao grosso do trabalho, ter sido o HTK, considerou-se mais proveitoso proceder ao cálculo dos coeficientes MFCC com as ferramentas providenciadas pelo HTK, evitando-se assim possíveis problemas de concordância de formatos ou outros.

Para o cálculo dos coeficientes foi escolhida uma janela de Hamming[12] de 15 ms de duração, 14 filtros (com forma triangular) e passos (avanço da janela) de 10 ms para uma obter o *frame-rate* tradicionalmente usado de 100 tramas por segundo. O número de coeficientes cepstrais usado foi 12, o que também é usual em processamento da fala. O uso de janelas mais curtas que o usual deve-se a proporcionar uma melhor adaptação à fala de crianças que têm tom mais alto.

Além dos 12 parâmetros MFCC e do coeficiente de ordem zero, c_0, os vectores de características associados a cada trama de sinal de fala têm também parâmetros ditos "dinâmicos". Estes correspondem a uma derivada temporal discreta (inclinação de regressão linear numa janela de 5 tramas) de cada componente do vector. São designados usualmente por parâmetros "DELTA". A variação dos parâmetros DELTA (parâmetros de aceleração) é também incluída. Desta forma os vectores de parâmetros têm dimensão 39 (13 coeficientes MFCC, 12+c_0, mais 13 coeficientes DELTA e mais 13 coeficientes de aceleração).

Em anexo apresenta-se o ficheiro `'config_Hcopy.txt'`, com os parâmetros de configuração da ferramenta `HCopy` do HTK. Esta ferramenta `HCopy` gera os ficheiros com os parâmetros MFCC a partir dos ficheiros áudio e das etiquetas correspondentes no ficheiro MLF[13].

O conjunto de ficheiros MFCC correspondentes às locuções de fala foi dividido em dois conjuntos: um conjunto de 'treino' e de 'teste' dos modelos de fones. A divisão foi de quatro quintos para o conjunto de 'treino' e o restante um quinto para o conjunto de 'teste'. O script '`mfclist.m`' gerou as duas listagens de ficheiros de treino e de teste.

[10] Software (freeware) concebido por Paul Boersma e David Weeninkpara para análise de fala em termos fonéticos.
[11] Representa o grau de periodicidade acústica, também chamado de Harmonics-to-Noise Ratio (HNR).
[12] O 'cosseno-elevado' desta janela serve para minimizar as laterais do lobo, enfatizando a zona central.
[13] Paramatrização indicada no ficheiro de configuração (Anexo A - 7.1) como:
TARGETKIND = MFCC_0_D_A

O ficheiro MLF foi seguidamente separado em dois novos ficheiros MLF, um com as transcrições e outro contendo apenas as etiquetas extralinguísticas, através do *script* `'cutmlf.m'`. Paralelamente à criação dos dois MLF, foram também gerados através do mesmo *script* novos ficheiros de MFCC de acordo com os ficheiros separados MLF e as suas respectivas marcações temporais das etiquetas.

Optou-se por esta abordagem de separar as locuções da restante componente não-falada dos ficheiros de parâmetros em diferentes ficheiros MLF e, por conseguinte, em diferentes grupos de ficheiros MFCC, devido à elevada complexidade na sequência da grande maioria dos ficheiros de áudio. A generalidade das sequências possuíam padrões consideravelmente mais complexos do que o normal ' <etiqueta> + palavra + <etiqueta> '. Durante as gravações de áudio, as crianças por diversas vezes hesitavam ou alteravam o que estavam a dizer a meio da locução, riam audivelmente ou respiravam pesadamente e, somente em raras ocasiões, diziam unicamente a palavra requerida, resultando assim em anotações com sequências de etiquetas das mais variadas e estranhas formas.

Uma vez cortados os ficheiros, procedeu-se à criação de grupos de 'treino' e de 'teste' para os modelos, a partir dos recentemente gerados ficheiros MLF. Assim, dois novos ficheiros foram criados contendo uma lista dos ficheiros MFCC que correspondem às etiquetas existentes no ficheiro MLF. O script `'mfclist.m'` gerou os dois referidos ficheiros, dividindo o número total de etiquetas no MFL e atribuindo quatro quintos dos correspondentes MFCCs à lista de 'treino' e o restante quinto como uma lista de 'teste'.

4.2. Sistema de Reconhecimento

4.2.1 Treino dos Modelos

O processo de criação e treino dos modelos de fones seguidamente descrito, foi baseado e adaptado do *tutorial* descrito no 'HTK Book 3.4' [35] e as instruções presentes na página da VoxForge [42].

A informação contida nas etiquetas dos ficheiros MLF são palavras. Considerando o objectivo do presente estudo de avaliar a correcta articulação de palavras na sua sequência de fones, tornou-se necessário transcrever as palavras nos seus fones constituintes. Para isso bastou usar o

dicionário e manipular o ficheiro de etiquetas. Assim, dando o ficheiro MLF, foi criado uma nova versão do mesmo mas, desta feita, com um fone por linha e dividindo os tempos (do início ao fim da palavra) em partes iguais por cada fone. Foi ainda gerada uma outra versão do mesmo ficheiro, também com um fone por linha, mas sem indicações temporais. Isto foi feito usando a ferramenta `HLEd` do HTK. Na figura seguinte mostra-se um exemplo do resultado da aplicação desta ferramenta. Neste caso, a ferramenta tomou a locução do ficheiro como 'copo' e procurou a sua tradução em SAMPA no ficheiro dicionário, gerando então o novo MLF com a tradução e dividindo equitativamente os tempos atribuídos a cada fone.

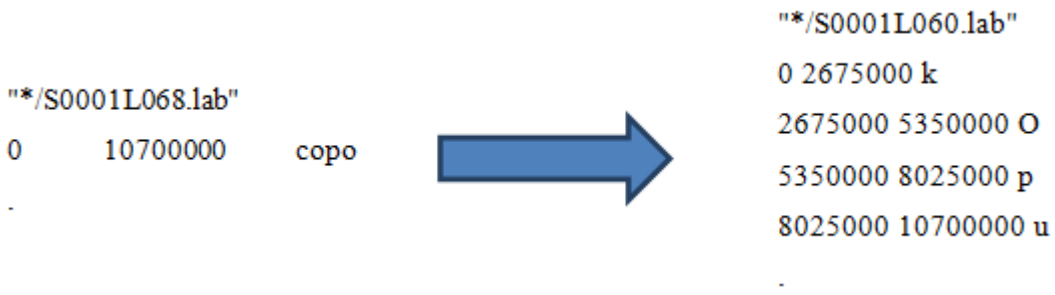

Figura 4. 7 – Exemplo do uso da ferramenta HLEd para a palavra 'copo' que tem a transcrição SAMPA 'kOpu'.

O passo anterior é essencial para iniciar o processo de treino, uma vez que os modelos de reconhecimento escolhidos para este estudo são modelos de fones. Os modelos de fones iniciais foram gerados usando uma estratégia de *flat-start*, onde todos os modelos são iguais na fase inicial. Significa que todos os modelos de fones são iniciados com vectores de médias e matrizes de covariância iguais à média e variância global de todo o material acústico de treino. Dá-se o nome de protótipo (`'proto'`) a este modelo inicial, que também tem definida a topologia esquerda-direita dos modelos com 3 estados. Este '*flat-start*' é feito usando a ferramenta `HCompV`.

A partir destes novos modelos de fones, foi gerado um ficheiro MMF (*Master Macro File*) que é simplesmente um ficheiro com o conjunto de todos os modelos criados.

Uma vez criado este ficheiro MMF procedeu-se à re-estimação dos modelos em sete passos. Para tal foi usada a ferramenta `HERest` que executa o algoritmo Baum-Welch, re-estimando todo o conjunto de HMMs.

O passo seguinte consiste na escolha da pronunciação mais adequada sempre que existem variações de pronúncia autorizadas no dicionário de pronunciações. Isso é feito forçando um alinhamento entre as observações acústicas e as sequências alternativas de fones

correspondentes. A ferramenta `HVite` em modo de alinhamento é usada para este efeito. O `HVite` faz uso do algoritmo de Viterbi (descrito na secção 3.2) para escolher qual o modelo (quando existe alternativas) que mais provavelmente gerou aquela sequência de observações. Como foi já anteriormente mencionado, este algoritmo tenta calcular o melhor caminho através de uma grelha onde uma dimensão corresponde aos possíveis estados, e outra às tramas (tempo).

Após o alinhamento, foram executadas mais duas re-estimações aos modelos.

O processo acima descrito encontra-se esquematizado na figura seguinte.

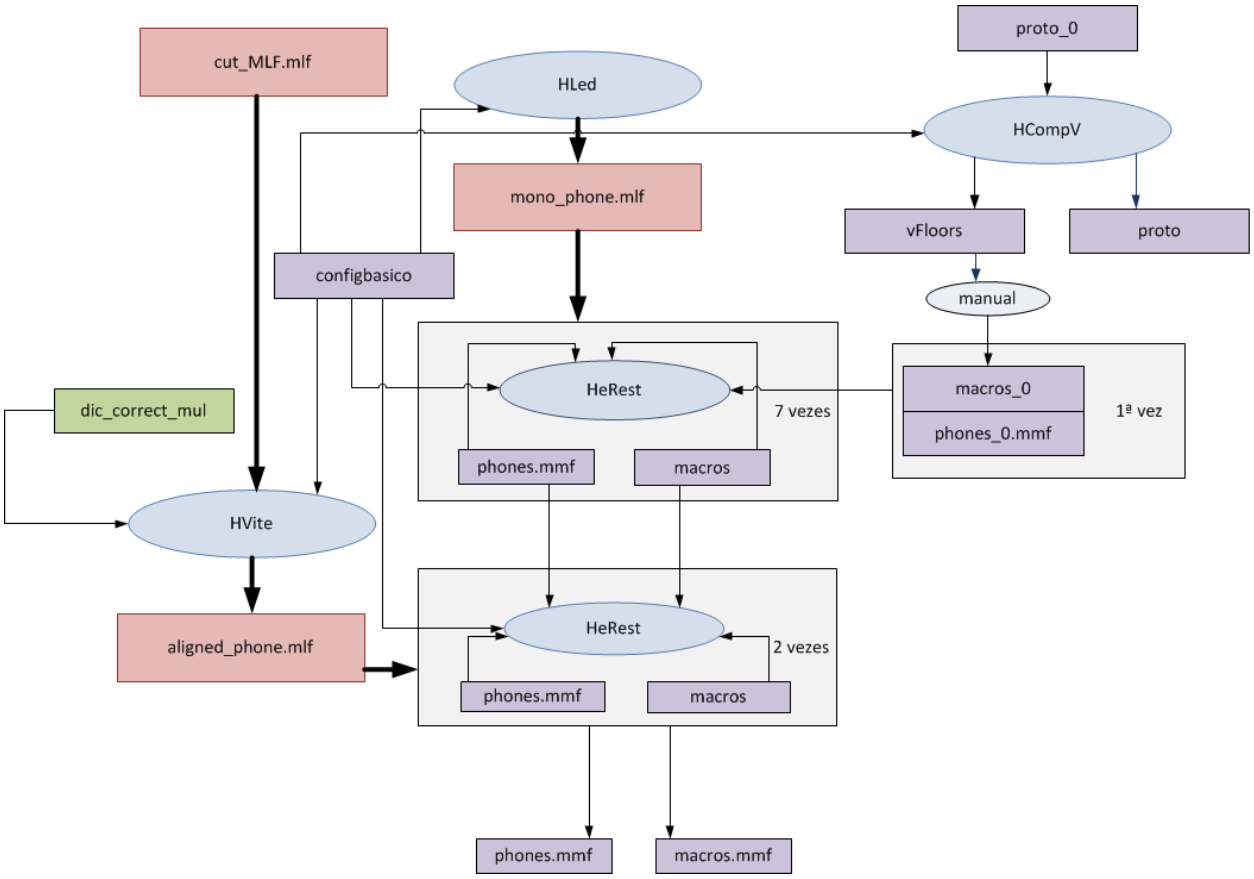

Figura 4. 8 – Diagrama do processo executado no HTK para treino dos modelos de fones. As elipses indicam ferramentas ou processos, sendo que a maioria são ferramentas do HTK estando também presente um processo manual. Os ficheiros MLF encontram-se representados a cor-de-rosa, enquanto os ficheiros utilizados pelo HTK se encontram representados a lilás. O dicionário com a lista de palavras correctas e sua correspondente transcrição em SAMPA está representado a verde. Este diagrama representa de uma forma concisa o funcionamento do script `'treino.bat'` (secção de anexos 7.3).

O passo seguinte consiste no incremento das componentes Gaussianas das funções densidade de probabilidade de cada estado (GMMs). O número de Gaussianas foi incrementado fazendo uso da ferramenta HHEd que executa um processo chamado *mixture splitting* quando recebe como

argumento "MU". Este processo foi executado quinze vezes para todos os modelos, fazendo três re-estimações entre cada incremento, de forma a obter GMMs com 16 componentes (secção de anexos 7.3).

4.2.2 Teste dos Modelos

Com os modelos devidamente treinados, torna-se necessária uma avaliação preliminar da taxa de reconhecimento do sistema. Para efectuar reconhecimento, o sistema necessita conhecer *a priori*, as palavras que pode reconhecer, os seja, uma gramática da qual faz parte uma lista de todas as palavras permitidas e as sequências autorizadas. Para tal, faz uso de uma "gramática" e da "rede" correspondente.

Para gerar a gramática, uma lista de palavras foi criada manualmente a partir da lista de todas as locuções recolhidas das etiquetas do MLF, seleccionando apenas as palavras correctas ou bem pronunciadas (lista igual à constante do ficheiro de dicionário `'dic_correct_mul'` presente no anexo 7.2). Esta lista foi atribuída à variável `'$word'` na figura seguinte. Também foi criada uma lista com as palavras/expressões de ligação mais frequentes, atribuída à variável `'$garbage'` na mesma figura. As sequências autorizadas são aquelas indicadas pelo grafo (ou rede) da figura, por exemplo, "Ahm um copo", onde "ahm" e "um" estão incluídos nos dois nodos opcionais do grafo. De notar que uma palavra do vocabulário tem de ocorrer uma e uma só vez, e sempre na última posição da sequência gerada.

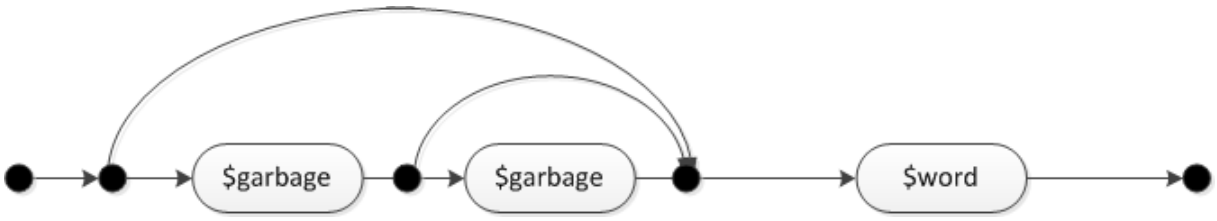

Figura 4. 9 – Diagrama da sequência autorizada no reconhecimento.

Com esta gramática, a ferramenta `HParse` pode então gerar o ficheiro de rede no formato SLF (Standard Lattice Format). Este formato serve para fornecer ao reconhecedor as múltiplas hipóteses de ocorrência de etiquetas e possui um conjunto de nós que representam as palavras e um conjunto de arcos que representam as possíveis transições entre palavras.

Fazendo uso dos modelos treinados, da lista de fones e da rede criada no passo anterior, o `HVite` pode então ser usado para efectuar o reconhecimento dos ficheiros de 'teste'. Assim, o

`HVite` pode gerar um ficheiro MLF de reconhecimento ('REC') com as sequências reconhecidas de acordo com a gramática. O ficheiro gerado no reconhecimento poderá ser comparado ao MLF com as transcrições reais para efeitos de avaliação de desempenho.

O `HResults` é uma ferramenta de análise de desempenho. Compara ficheiros de etiquetas, por norma o ficheiro de *output* produzido pelo `HVite`, com um ficheiro de referência das transcrições, calculando as estatísticas da comparação como a correcção e a precisão. Para o efeito usa um procedimento de alinhamento de *strings*. Um exemplo do seu *output* pode ser observado na figura seguinte.

```
====================== HTK Results Analysis =======================
  Date: Sat Sep 08 19:57:27 2012
  Ref : MLF_cut2_lix.mlf
  Rec : ..\test\recout_FINAL.mlf
------------------------ Overall Results --------------------------
SENT: %Correct=79.66 [H=423, S=108, N=531]
WORD: %Corr=78.58, Acc=78.58 [H=433, D=20, S=98, I=0, N=551]
====================================================================
```

Figura 4. 10 – Exemplo do *output* da análise estatística produzida pela ferramenta.

Nesta avaliação do desempenho dos modelos foi obtida uma taxa de correcção de 78.58% com 433 palavras reconhecidas correctamente (H - hits), 20 apagamentos (D - *deletions*), 0 inserções (I - *insertions*) e 98 substituições (S – *substitutions*), num total de 551 palavras reconhecidas. Em grande parte dos casos onde ocorreram erros de reconhecimento, a palavra dada como output é semelhante à palavra do MLF de referência (com o que foi realmente dito), como se pode constatar em alguns dos exemplos presentes na figura seguinte.

```
Aligned transcription: S0098L082.lab vs S0098L082.rec
 LAB: garrafa vinho
 REC:         passarinho
Aligned transcription: S0053L103.lab vs S0053L103.rec
 LAB: livro
 REC: livros
 Aligned transcription: S0089L178.lab vs S0089L178.rec
 LAB: talhere
 REC: talheres
 Aligned transcription: S0078L079.lab vs S0078L079.rec
 LAB: fralda
 REC: fada
 Aligned transcription: S0061L120.lab vs S0061L120.rec
 LAB: mola
 REC: mala
```

Figura 4. 11 – Exemplo de erros de reconhecimento (substituições) presentes no ficheiro de *output* do `HResults`.

O MLF gerado segundo esta gramática possui apenas uma etiqueta (que será obrigatoriamente uma única palavra da lista de palavras consideradas correctas), por cada transcrição. O `HVite` ignora todos os resultados da lista `$garbage`, não os incluindo no ficheiro MLF gerado (REC).

De relevante importância é ainda o facto de se ter adoptado a sequência da gramática apresentada na Figura 4. 9. A escolha desta gramática permite um reduzido peso computacional no reconhecimento porque diminui o número de possíveis sequências (arcos). No entanto, o custo associado é a redução da taxa de palavras correctamente reconhecidas. Como foi mencionado anteriormente, muitas das transcrições apresentavam sequências de etiquetas invulgares, existindo mesmo algumas com mais do que uma etiqueta com palavras consideradas correctas. O resultado global do reconhecimento sofre assim, em certo modo, visto estas transcrições não poderem ser correctamente reconhecidas com a gramática escolhida.

5. Verificação

Uma vez criado o sistema de reconhecimento de fones, pode então ser gerado o procedimento para avaliação e verificação das palavras proferidas de acordo com o que era esperado. Um estudo com objectivos semelhantes foi desenvolvido para a linguagem sueca. Foi proposta a criação de um sistema para segmentação e alinhamento automático de fala obtendo resultados que, apesar de promissores, revelam ainda bastantes falhas em termos da correcção do alinhamento produzido. [43]

Trata-se de um problema de teste de hipóteses e consiste em saber se o locutor disse ou não aquilo que lhe era pedido para dizer. Posteriormente, e em caso negativo, é também importante saber quais os fones que foram mal pronunciados.

O `HVite`, ao gerar o ficheiro de reconhecimento, pode também fornecer a informação das verosimilhanças associadas a cada palavra ou fone reconhecido. Este valor do logaritmo da verosimilhança de palavra calculado pelo `HVite` será referido deste ponto em diante como L_1. De notar o facto de este valor de L_1 ser negativo, portanto, quanto mais próximo de zero for esse valor, maior será a verosimilhança calculada.

Como medida de desempenho para o teste de hipóteses é usada uma diferença de verosimilhanças obtidas com o algoritmo de Viterbi. A primeira dessas verosimilhanças, L_1, é aquela obtida com o reconhecedor segundo a gramática da Figura 4. 9. A outra verosimilhança, L_2, é a obtida através da gramática representada pela figura seguinte.

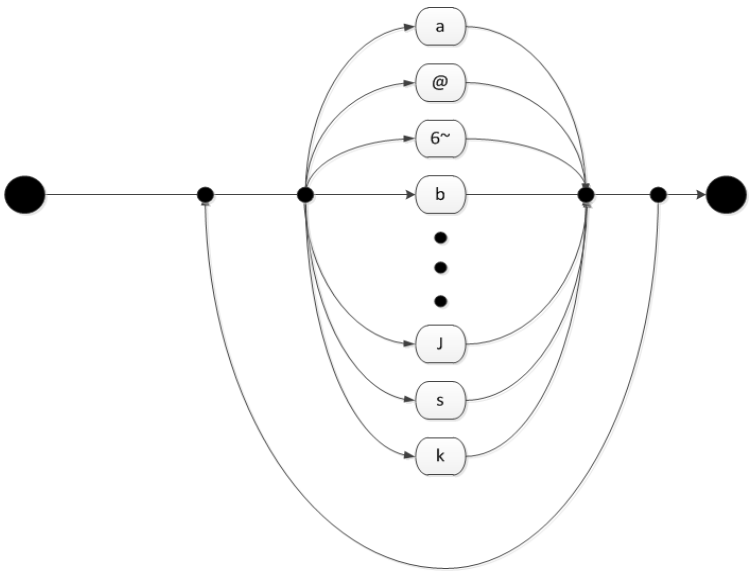

Figura 5. 1 – Diagrama da sequência livre de fones autorizada pela gramática livre.

Para a identificação e análise de disfluências e incorrecções nas locuções considerou-se que a estratégia mais vantajosa a adoptar seria a da utilização de uma gramática 'livre' (por oposição à gramática descrita na secção 4.2.2, na Figura 4. 9, a qual passará a ser referida como gramática 'fixa'), onde o reconhecedor pudesse reconhecer uma e uma só palavra, com uma qualquer sequência de fones. A figura anterior descreve esta gramática 'livre'. Assim, a sequência de fones gerada desta forma produz uma verosimilhança, L_2, sempre maior ou igual a L_1 uma vez que a sequência de fones é óptima segundo os modelos.

Se a diferença de verosimilhanças L_2–L_1 for baixa significa que a palavra foi bem pronunciada. A sequência de fones não tem de ser exactamente igual; podem até existir repetições, apagamentos ou inserções de alguns fones. No caso de uma diferença elevada, pode significar que o locutor não disse a palavra que era suposto dizer.

Para proceder à avaliação da veracidade da premissa criou-se um script de Matlab que analisasse todos os ficheiros MFCC contemplados no conjunto de teste, comparando o resultado obtido pelo reconhecedor com a o que foi inicialmente solicitado ao locutor (criança), imprimindo as informações relevantes num ficheiro. Temos assim 3 etiquetas potencialmente diferentes:

- REF: a palavra solicitada à criança;
- SAID: a palavra que a criança disse (possivelmente uma palavra mal pronunciada);
- REC: a palavra reconhecida automaticamente pelo sistema.

Assim, numa primeira instância, considerou-se que a criança disse exactamente o que lhe foi pedido (MLF de referência igual ao MLF das palavras a dizer), fazendo o reconhecedor passar pelas duas gramáticas anteriormente descritas (gramática 'fixa' e gramática 'livre') e comparando as verosimilhanças e as sequências de fones obtidas pelos dois processos.

REF	SAID	REC	phones	Freephones	Nframes	L1	L1-L2	align_score	refph_algn	freeph_algnt
coelho	coelho	coelho	ku^{a}Lu	u~fboa^{a}~Lu	94	-5.245491e+001	-1.3038e+000	6	ku6···Lu	üfboaãLu
caixa	caixa	caixa	kaiS^{a}	kaS^{a}	59	-5.522873e+001	-350.1587e-003	1	kai$S$6	ka·S6
anel	anel	anel	^{a}nEl	^{a}nE^{a}~gr	49	-6.122777e+001	-30.1912e-003	3	6nEl··	6nEãgr
ovelha	ovelha	pilha	Ov^{a}L^{a}	O^{a}fiL^{a}	77	-5.395169e+001	-1.1648e+000	3	O·v6··L6	O6fiL6
lâmpada	lâmpada	bola	l^{a}~p^{a}d^{a}	poObld^{a}~^{a}	113	-5.809059e+001	-1.1063e+000	6	lãp6·d·6	poObldã6
prato	prato	prato	pratu	pratu	44	-6.170283e+001	0.0000e+000	0	pratu	pratu
tomate	tomate	tomate	tumat@	RmaGt@	68	-5.971878e+001	-312.9497e-003	3	tuma·t@	·RmaGt@

Figura 5. 2 – Excerto do ficheiro gerado que compara as diferentes verosimilhanças e as *strings* de fones para ficheiros REF e SAID iguais.

O ficheiro gerado, cujo excerto pode ser observado na figura anterior, contém informação relevante a cada etiqueta:

- As transcrições nos diferentes ficheiros MLF: REF, SAID e REC (o que foi pedido à criança, o que ela de facto disse e o que o `HVite` reconheceu, respectivamente);
- A sequência de fones correspondente à palavra pedida segundo o dicionário (phones) e a sequência de fones dada pelo reconhecedor com recurso à gramática 'livre' (Freephones);
- O número de *frames* correspondente à palavra (Nframes);
- O valor de verosimilhança para a gramática com a tradução SAMPA da palavra – L_1;
- O valor da diferença entre a verosimilhança anterior e o obtido com uma gramática livre: L_1-L_2;
- Uma medida (align_score) de alinhamento entre as duas *strings* de fones: a sequência de fones da palavra e a sequência de fones 'livre'. O alinhamento usa o algoritmo 'edit distance'[14].

Os valores de L_1 e L_2 apresentados no ficheiro representam a soma dos valores de verosimilhança calculados pelo `HVite` para cada fone da palavra em questão, normalizada pelo número de *frames* (Nframes).

No caso do uso da gramática 'fixa', o reconhecedor é forçado a calcular o caminho de Viterbi por uma sequência de fones específica, de modo a que corresponda exactamente à sequência ditada pela tradução SAMPA da palavra presente no dicionário. Se for proferido um fone não esperado na sequência, a verosimilhança será obrigatoriamente menor, isto é, um valor de L_1 menor.

Já no caso da gramática 'livre', o reconhecedor gerará uma sequência de fones com os modelos mais parecidos com os parâmetros em análise e portanto, um caminho óptimo segundo os modelos.

[14] *Script* de Matlab desenvolvido no Laboratório de Processamento de Sinal que compara duas *strings* atribuindo pontos por cada inserção, supressão ou troca de caracter. O resultado obtido como *output* será tanto maior quanto maiores forem as diferenças entre as *strings*.

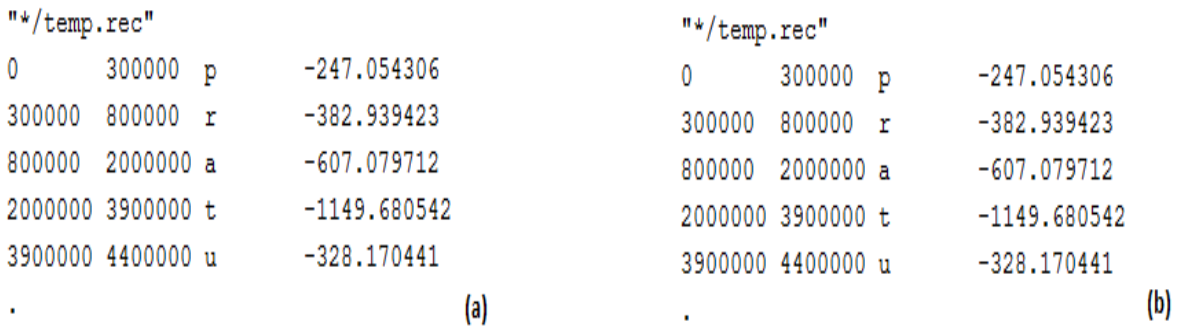

Figura 5. 3 – Resultados produzidos pelo `HVite` para o ficheiro 'S0054L152' com recurso à gramática 'fixa' (a) e à gramática 'livre' (b), para ficheiros REF e SAID iguais.

No caso demonstrado na figura anterior, os parâmetros obtidos da locução da criança encontram-se perfeitamente enquadrados com a sequência de fones esperada para aquela palavra ('prato'). Esse facto, conjuntamente com um valor total (do somatório) de verosimilhanças relativamente baixo, permite inferir que a articulação da palavra foi correcta.

No exemplo da figura seguinte é possível verificar que a dicção da palavra ('coroa') não foi perfeita, indicando o reconhecedor com a gramática 'livre' uma supressão (apagamento) do fone 'u' e uma inserção do fone 'a'.

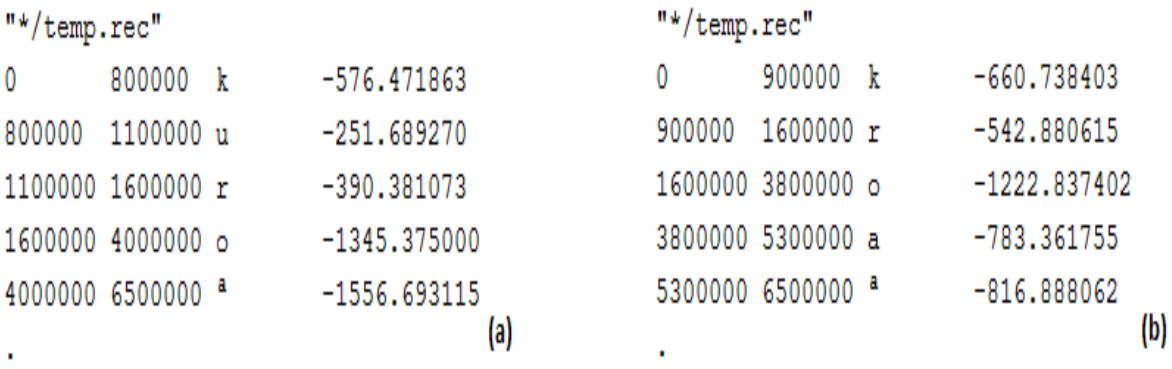

Figura 5. 4 – Resultados produzidos pelo HVite para o ficheiro 'S0097L000' com recurso à gramática 'fixa' (a) e à gramática 'livre' (b), para ficheiros REF e SAID iguais.

Por análise dos resultados compilados no ficheiro, é possível observar que, como expectável, os valores de L_2 são sempre iguais ou superiores aos de L_1. Também se verifica uma tendência para que os menores valores no *score* do alinhamento (sequências com menos necessidade de alterações no alinhamento) correspondam a valores maiores de L_2 (maior verosimilhança). É também notória uma diminuição nos valores de L_1 e L_2 (menor verosimilhança) com o aumento do número de fones e complexidade da palavra.

Posteriormente procedeu-se à mesma análise mas simulando que a criança não disse o que lhe foi pedido (os dados fornecidos para reconhecimento não correspondem ao que se diz ao reconhecedor que a criança disse). Para tal, gerou-se um ficheiro MLF a partir do REF, onde as transcrições foram alteradas de modo aleatório por outras palavras constantes do dicionário. Para o efeito foi criado o *script* `'changeMLFlab.m'`.

REF	SAID	REC	phones	Freephones	Nframes	L1	L1-L2	align score	refph algn	freeph algnt
cereais	coelho	coelho	s@riai8	u~fboa^{a}~Lu	94	-5.979505e+001	-8.6440e+000	7	s@riai8·	üfboaãLu
gomas	caixa	caixa	gom^{a}8	ka8^{a}	59	-6.165912e+001	-6.7806e+000	4	gom68	ka86·
cubo	anel	anel	kubu	^{a}nE^{a}~gr	49	-7.215646e+001	-10.9589e+000	6	kubu··	6nEãgr
cenoura	ovelha	pilha	s@nor^{a}	O^{a}fiL^{a}	77	-6.251457e+001	-9.7277e+000	5	s@nor6	O6fiL6
chavina	lâmpada	bola	S^{a}vin^{a}	poObld^{a}~^{a}	113	-6.204074e+001	-5.0565e+000	7	S6vin··6	poObldã6
maçã	prato	prato	m^{a}s^{a}~	pratu	44	-6.562010e+001	-3.9173e+000	5	m6sã·	pratu
retrato	tomate	tomate	R@tratu	RmaGt@	68	-6.345029e+001	-4.0445e+000	5	R@tratu	RmaG·t@

Figura 5. 5 – Excerto do ficheiro gerado que compara as diferentes verosimilhanças e as *strings* de fones para ficheiros REF e SAID diferentes.

As informações presentes no ficheiro gerado permitiram verificar uma diminuição generalizada (mais negativos) nos valores de L_1 (e por consequência na diferença L_1–L_2) e nos *scores* do alinhamento, o que corrobora a tese de que poderá ser possível identificar disfluências com base nos valores de verosimilhança.

No exemplo da figura seguinte é possível observar os resultados do reconhecedor, segundo as duas gramáticas, para uma etiqueta onde a palavra pedida (REF toma o valor 'carta') foi alterada de modo a não coincidir com o que a criança realmente disse (SAID) que, neste exemplo, foi a palavra 'praia'.

```
"*/temp.rec"
0       1300000 k       -1041.198120
1300000 2600000 a       -731.584534
2600000 4000000 r       -945.395264
4000000 4300000 t       -288.709442
4300000 4900000 ª       -428.117767
.                                    (a)
```

```
"*/temp.rec"
0       400000  p       -286.109436
400000  1400000 r       -700.334595
1400000 2600000 a       -667.383606
2600000 4100000 i       -927.261780
4100000 4900000 ª       -550.425964
.                                    (b)
```

Figura 5. 6 – Resultados produzidos pelo HVite para o ficheiro 'S0097L000' com recurso à gramática 'fixa' (a) e à gramática 'livre' (b), para etiquetas REF e SAID diferentes.

Procedeu-se então à análise ROC (*Receiver operating characteristic*) dos valores de L_1–L_2 para se poder escolher um limiar de aceitação mais adequado ao problema de classificação em análise – 'Aceitar ou não a locução como correcta.'. Em detecção de sinais, as curvas ROC são gráficos de descrição de um sistema de classificação binária e do seu desempenho [44].

Considerou-se então que a hipótese H0 é: 'A criança disse correctamente o que lhe foi pedido.', havendo quatro possíveis soluções, em função da resposta do reconhecedor, como demonstra o quadro da figura seguinte:

- Correctamente aceite (ou *True Positive* – TP) - se a hipótese H0 foi aceite e estava certa;
- Correctamente rejeitado (ou *True Negative* – TN) - se a hipótese H0 não foi aceite e estava errada;
- Falsamente aceite (ou *False Positive* - FP) - se a hipótese H0 foi aceite e estava errada;
- Falsamente rejeitado (ou *False Negative* - FN) - se a hipótese H0 não foi aceite e estava certa.

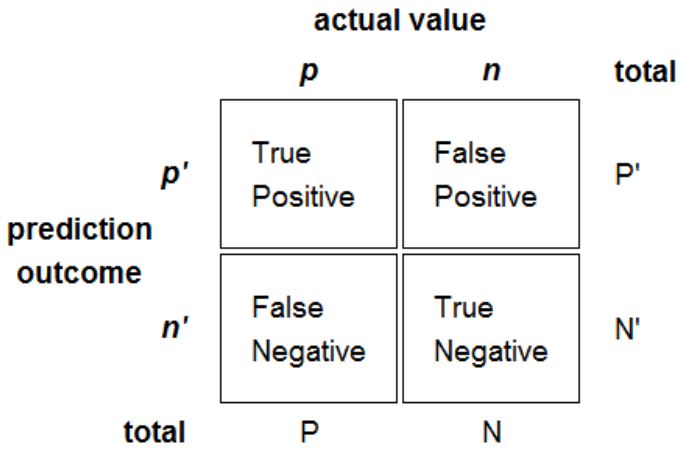

Figura 5. 7 – Exemplo da matriz de confusão 2x2 que representa o corrente problema de classificação. [44]

Idealmente, o reconhecedor deveria produzir o maior número de correctamente aceites e correctamente rejeitados e um número mínimo de falsamente aceites e falsamente rejeitados, minimizando assim os erros de reconhecimento. Em termos práticos, a escolha do limiar de decisão não é mais que um compromisso que torna o reconhecedor mais ou menos flexível na correcção (benefícios), em detrimento do número de erros (custos).

Assim, são usadas diversas medidas, cujos valores podem ser inferidos directamente da matriz de confusão, das quais se destacam:

- *Sensitivity* ou *True Positive Rate* - $TPR = TP / P = TP / (TP + FN)$;
- *False Positive Rate* - $FPR = FP / N = FP / (FP + TN)$;
- *Accuracy* - $ACC = (TP + TN) / (P + N)$.

A escolha do limiar de decisão/aceitação deve ser tomada dependendo de qual a medida a atribuir maior relevância. A prioridade a atribuir à minimização dos resultados falsamente aceites (falsos positivos) ou à redução dos falsamente rejeitados (falsos negativos) dita a localização do limiar, visto serem medidas de crescimento inversamente proporcional.

Trata-se portanto, de uma decisão de estratégia de reforço psicológico positivo ou negativo, pesando se será mais profícuo detectar mais erros (correctamente rejeitados), implicando isso um maior número de casos rejeitados em que a criança de facto disse a palavra correcta (falsamente rejeitados), ou, adoptando a estratégia inversa, maximizar o número de correctas aceitações, acarretando um aumento no número de falsas aceitações.

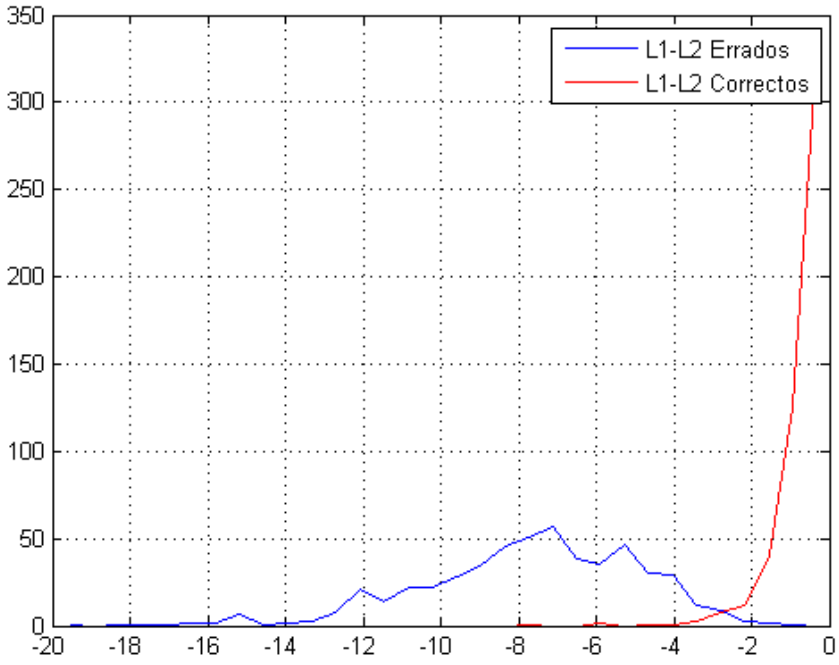

Figura 5. 8 – Gráfico das curvas que compara a distância L1-L2 para as transcrições correctas e as erradas.

O traçado das duas curvas da diferença L_1-L_2 para as palavras correctamente pronunciadas e para as palavras erradas permite observar as verosimilhanças para os dois casos, dando uma boa perspectiva da sua distribuição. O ponto de intersecção das duas curvas ROC é um bom indicador de onde se deverá localizar um bom limiar.

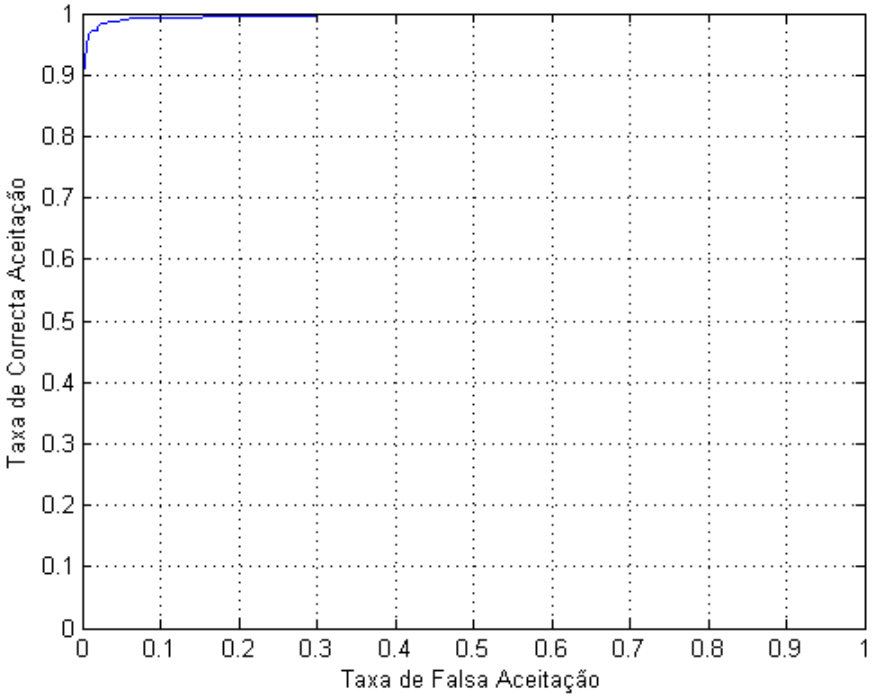

Figura 5. 9 – Curva ROC

A curva ROC relaciona a Taxa de correctas aceitações com a taxa de falsas aceitações, sendo que o seu ponto ideal de operação se encontra nas coordenadas (0,1), onde todas as correctas aceitações seriam bem detectadas e não produzindo nenhum falso alarme (falsa aceitação).

No entanto, para melhor proceder à avaliação do problema, uma comparação do compromisso entre os dois tipos de erro, revela-se mais eficaz.

Para tal, recorreu-se também a gráficos DET (*Detection Error Tradeoff*) [45] cuja curva representa a relação da taxa de falsas rejeições em função da taxa de falsas aceitações (compromisso entre os diferentes tipos de erro). Este gráfico produz curvas mais lineares que as curvas ROC, conseguindo um melhor aproveitamento da área da imagem para a região crítica de operação.

Considerando ambas as taxas de erro com o mesmo peso, a solução ideal para a optimização do problema seria o ponto no gráfico DET com a menos distância à origem do referencial.

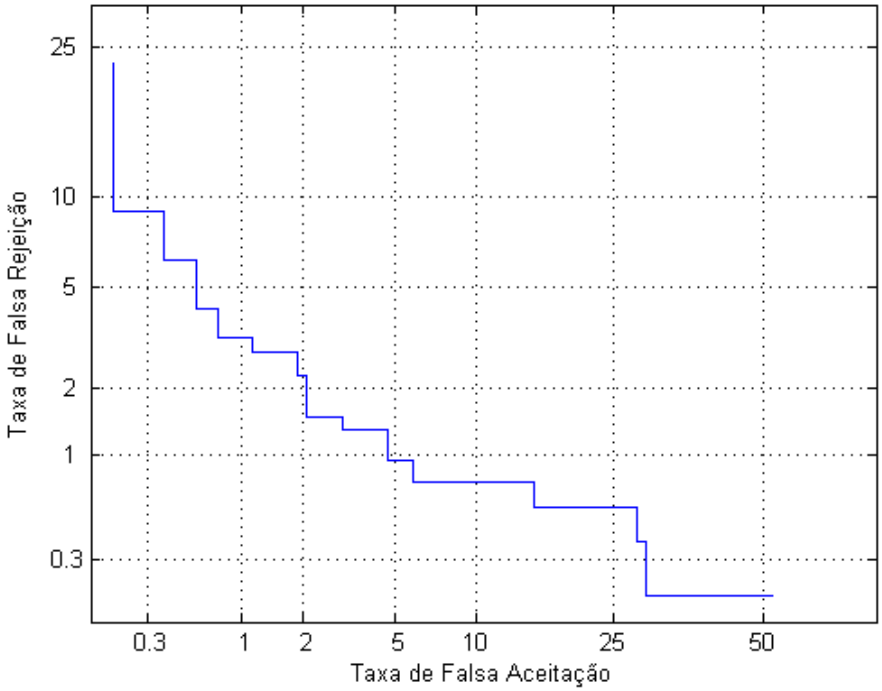

Figura 5. 10 – Curva DET.

Para o efeito deste estudo, e tendo em conta que é direccionado a crianças, será mais interessante minimizar o número de falsas rejeições (a criança disse correctamente a palavra mas o sistema reconheceu-a como erro), permitindo comprometer em certo ponto a taxa de falsas aceitações (erros da criança aceites pelo sistema como correctos). Assim, um limiar correspondente a 5% de falsa aceitação contra 0.9% de falsa rejeição ou 10% de falsa aceitação contra 0.8% de falsa rejeição poderão ser uma boa escolha.

Como foi já referido, o sistema não é excepcionalmente preciso ao nível do reconhecimento de fones individuais, podendo mesmo apresentar inserções ou substituições (em relação à sequência esperada da palavra) que podem ser ignoradas sempre que os seus valores de verosimilhança sejam relativamente baixos. Para tornar o sistema mais robusto é possível usar os valores '`align_score`' calculados como uma segunda medida de classificação.

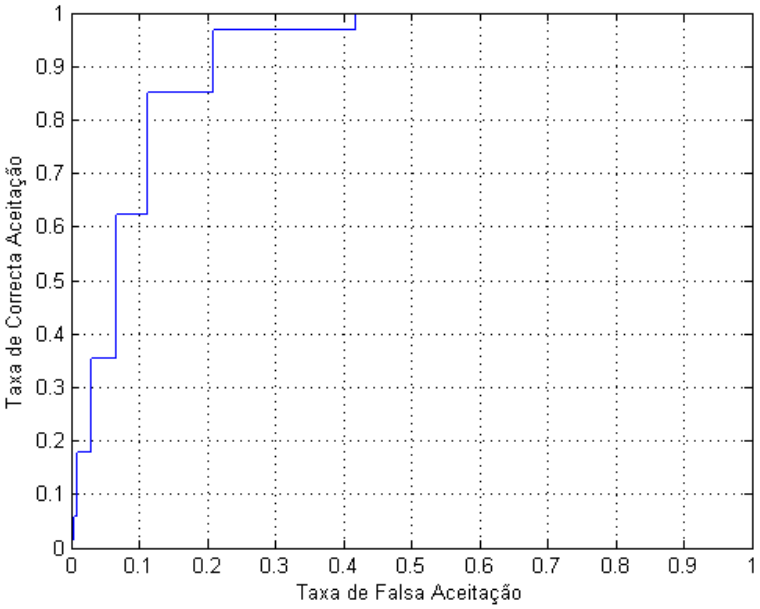

Figura 5. 11 – Curva ROC para o *score* de alinhamento.

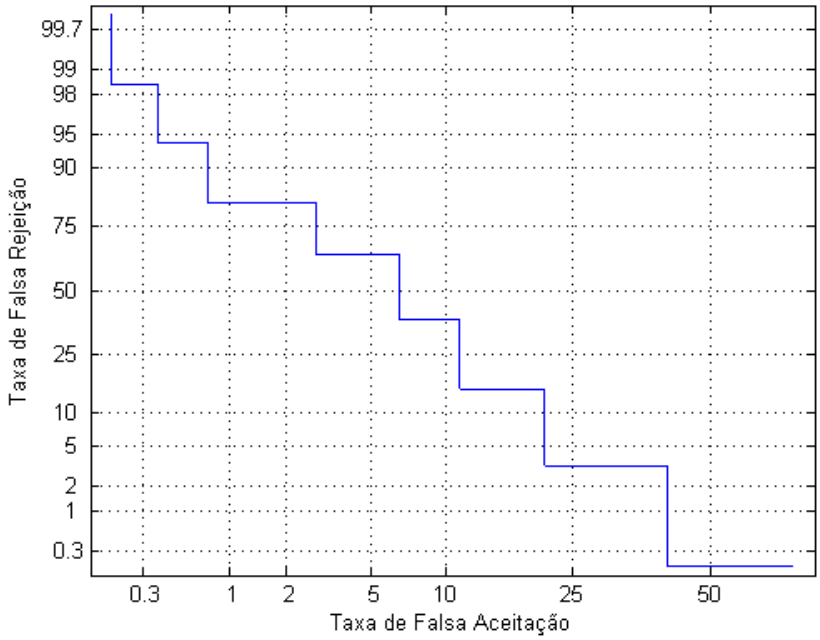

Figura 5. 12 – Curva DET para o *score* de alinhamento.

A Figura 5. 11 e a Figura 5. 12 representam a curva ROC e a curva DET, respectivamente, para os valores de *score* de alinhamento, à imagem do que foi feito com os valores de verosimilhança.

6. Conclusão e Trabalho Futuro

Este estudo tinha como objectivo principal a criação de um sistema de identificação de disfluências. Este objectivo foi implementado com sucesso. Foi também desenvolvido um sistema de avaliação da correcção da dicção das palavras proferidas em função dos valores de verosimilhança calculados com os modelos de fones. A decisão baseada nesta medida mostra um bom compromisso entre a taxa de falsas aceitações e falsas rejeições. A decisão pode ainda ser melhorada com base nos valores de alinhamento de *strings* de fones calculados com base no reconhecimento de fones sem restrições e com a transcrição fonética da palavra esperada em questão. Foi ainda aplicado um método de classificação binária e sugeridos valores para os limiares de decisão.

Com base no presente trabalho é agora possível desenvolver o sistema global de auxílio à terapia da fala em crianças, tal como foi descrito na secção 2.3. Com efeito, esta tese constitui o núcleo tecnológico base que torna possível e útil o referido sistema. Para tal dever-se-á aplicar os modelos e o método de verificação desenvolvidos, embebidos no sistema que usará jogos ou outras actividades lúdicas na interacção com a criança.

Seria ainda interessante melhorar os modelos de reconhecimento através da recolha de mais locuções para compor a base de dados. Uma maior amostra de locuções tornará o sistema ainda mais robusto. O próprio método de identificação das disfluências pode também ser melhorado, por exemplo, aplicando alguns conhecimentos de linguística sobre a evolução das capacidades de fala em crianças. O estudo [46] pode servir de fundamento para as referidas implementações.

7. Anexos

7.1. Anexo A – Ficheiro de Configuração do HCopy

config_Hcopy.txt

```
NATURALREADORDER = TRUE
NONUMESCAPES = TRUE
NATURALWRITEORDER = TRUE

# Coding parameters
SAVECOMPRESSED = F
SAVEWITHCRC = F
SOURCEFORMAT=WAV
TARGETKIND = MFCC_0_D_A
TARGETRATE = 100000.0
WINDOWSIZE = 150000.0
USEHAMMING = T
PREEMCOEF = 0.97
NUMCHANS = 14
NUMCEPS = 12
ENORMALISE = F
```

7.2. Anexo B - Lista de Fones e Dicionário

Lista de fones com exemplo de uso

Fone	Exemplo de palavra
ᵃ	c**a**m**a**
ᵃ~	mel**an**cia
@	microfon**e**
a	p**a**to
b	**b**ola
d	rá**d**io
E	caram**e**lo
e	m**e**sa
e~	p**en**te
f	**f**aca
g	**g**olo
i	**i**greja
i~	**ín**dio
J	golfi**nh**o
k	**c**asa
l	**l**ápis
L	abe**lh**a
m	**m**ola
n	ca**n**eca

O	m**o**ta
o	m**o**sca
o~	b**om**beiro
p	**p**ato
r	ba**r**co
R	ca**rr**o
s	**c**enoura
S	e**s**pada
t	**t**oma**t**e
u	b**o**nec**o**
u~	**um**
v	**v**elho
Z	ca**s**a
z	quei**j**o

Dicionário de transcrição SAMPA - diccorrect_mul.txt

abacaxi	ª b a k a k s i
abelha	ª b ª L ª
abelha	ª b e L ª
abóbora	ª b O b u r ª
ananás	ª n ª n a S
anel	ª n E l
aranha	ª r ª J ª
arco-íris	a r k u i r i S
astronauta	ª S t r O n a u t ª
autocarro	a u t O k a R u
ave	a v @
avental	ª v e~ t a l
avioneta	ª v i u n E t ª
avião	ª v i ª~ u~
avô	ª v o
babete	b a b E t @
bailarina	b a i l ª r i n ª
balde	b a l d @
baleia	b ª l ª i ª
baliza	b ª l i z ª
baloiço	b ª l o i s u
balão	b ª l ª~ u~
banana	b ª n ª n ª
bananas	b ª n ª n ª S
bandeira	b ª~ d ª i r ª
banho	b ª J u
baralho	b ª r a L u
barco	b a r k u
barriga	b ª R i g ª
baton	b a t O n
bebé	b E b E
berbequim	b @ r b @ k i~
bicho	b i S u
bicicleta	b i s i k l E t ª
bicicletas	b i s i k l E t ª S
boca	b o k ª
bola	b O l ª

bolo	b o l u
bombeiro	b o~ b ª i r u
bombom	b o~ b o~
bombons	b o~ b o~ S
boneca	b u n E k ª
boneco	b u n E k u
borboleta	b u r b u l e t ª
bota	b O t ª
branco	b r ª~ k u
braço	b r a s u
brinquedo	b r i~ k e d u
bruxa	b r u S ª
burro	b u R u
cabelo	k ª b e l u
cabide	k ª b i d @
cachorro	k ª S o R u
cadeira	k ª d ª i r ª
caixa	k a i S ª
caixote	k a i S O t @
calça	k a l s ª
calças	k a l s ª S
calções	k a l s o~ i~ S
cama	k ª m ª
camelo	k ª m e l u
camioneta	k a m i u n E t ª
camisola	k ª m i z O l ª
camião	k a m i ª~ u~
candeeiro	k ª~ d i ª i r u
caneca	k ª n E k ª
caneta	k ª n e t ª
canguru	k ª~ g u r u
cangurus	k ª~ g u r u S
capacete	k ª p ª s e t @
caracol	k ª r ª k O l
caramelo	k ª r ª m e l u
caranguejo	k ª r ª~ g ª Z u
caranguejo	k ª r ª~ g ª i Z u
carrinha	k ª R i J ª
carro	k a R u
carrocel	k ª R O s E l
carta	k a r t ª
cartas	k a r t ª S
casa	k a z ª
cavalo	k ª v a l u
caçador	k ª s ª d o r
cebola	s @ b o l ª
cenoura	s @ n o r ª
cereais	s @ r i a i S
cereja	s @ r ª Z ª
cereja	s @ r ª i Z ª
cerejas	s @ r ª Z ª S
cerejas	s @ r ª i Z ª S
cerveja	s @ r v ª Z ª
cerveja	s @ r v ª i Z ª
cesta	s e S t ª
cesto	s e S t u
chapéu	S ª p E u

chave	S a v @
chavina	S ª v i n ª
chinelo	S i n E l u
chinelos	S i n E l u S
chocolate	S u k u l a t @
chupeta	S u p E t ª
chá	S a
cisne	s i Z n @
cobra	k O b r ª
coelho	k u ª L u
coelho	k u e L u
cofre	k O f r @
cogumelo	k u g u m E l u
colheres	k u L E r @ S
comboio	k o~ b O i u
computador	k o~ p u t ª d o r
copo	k O p u
coração	k u r ª s ª~ u~
corneta	k O r n e t ª
coroa	k u r o ª
cortina	k u r t i n ª
coruja	k O r u Z ª
cruzeta	k r u z e t ª
cubo	k u b u
cueca	k u E k ª
cuecas	k u E k ª S
cão	k ª~ u~
dado	d a d u
dedo	d e d u
dinossauro	d i n O s a u r u
direito	d i r ª i t u
doces	d o s @ S
doninha	d O n i J ª
dormir	d u r m i r
doutor	d o t o r
duas	d u ª S
elefante	e l @ f ª~ t @
escada	@ S k a d ª
escova	@ S k o v ª
espada	@ S p a d ª
esquerda	@ S k e r d ª
esquilo	@ S k i l u
estrela	@ S t r e l ª
estrelinha	@ S t r @ l i J ª
faca	f a k ª
fada	f a d ª
fato	f a t u
ferro	f E R u
figo	f i g u
flauta	f l a u t ª
flor	f l o r
foca	f O k ª
foguetão	f u g @ t ª~ u~
fogão	f u g ª~ u~
folha	f o L ª
formiga	f u r m i g ª
forno	f o r n u

fralda	f r a l d ª
fruta	f r u t ª
frutos	f r u t u S
galinha	g ª l i J ª
garfo	g a r f u
garfos	g a r f u S
garrafa	g ª R a f ª
gato	g a t u
gelado	Z @ l a d u
gelo	Z e l u
girafa	Z i r a f ª
golfinho	g o l f i J u
golfinhos	g o l f i J u S
gomas	g o m ª S
gravata	g r ª v a t ª
guitarra	g i t a R ª
helicóptero	e l i k O p t @ r u
hipopótamo	i p O p O t ª m u
homem	O m ª~ i~
igreja	i g r ª Z ª
igreja	i g r ª i Z ª
iogurte	i O g u r t @
janela	Z ª n E l ª
jarra	Z a R ª
jarro	Z a R u
joaninha	Z u ª n i J ª
lanterna	l ª~ t E r n ª
laranja	l ª r ª~ Z ª
lata	l a t ª
lego	l e g u
legos	l e g u S
legumes	l @ g u m @ S
leão	l i ª~ u~
limão	l i m ª~ u~
linha	l i J ª
livro	l i v r u
livros	l i v r u S
lixo	l i S u
lobo	l o b u
lua	l u ª
luva	l u v ª
luz	l u S
lápis	l a p i S
lâmpada	l ª~ p ª d ª
macaco	m ª k a k u
mala	m a l ª
mar	m a r
martelo	m ª r t E l u
maçã	m ª s ª~
melancia	m @ l ª~ s i ª
melão	m @ l ª~ u~
menino	m @ n i n u
mesa	m e z ª
microfone	m i k r O f O n @
milho	m i L u
mochila	m u S i l ª
mocho	m O S u

moinho	m u i J u
mola	m O l ª
morango	m u r ª~ g u
mosca	m o S k ª
mosquito	m u S k i t u
mota	m O t ª
mão	m ª~ u~
médico	m E d i k u
música	m u z i k ª
ninho	n i J u
nuvem	n u v ª~ i~
nuvens	n u v ª~ i~ S
olho	o L u
ovelha	O v ª L ª
ovelha	O v e L ª
ovo	o v u
ovos	O v u S
palhaço	p ª L a s u
panela	p ª n E l ª
papagaio	p ª p ª g a i u
pássaro	p a s ª r u
passarinho	p ª s ª r i J u
patim	p ª t i~
patins	p ª t i~ S
pato	p a t u
pedra	p E d r ª
peixe	p ª i S @
pendurar	p e~ d u r a r
pente	p e~ t @
penteado	p e~ t i a d u
pequena	p @ k e n ª
piano	p i ª n u
piscina	p i S s i n ª
pilha	p i L ª
pipocas	p i p O k ª S
pirata	p i r a t ª
piratas	p i r a t ª S
pizza	p i z ª
pião	p i ª~ u~
polícia	p u l i s i ª
pomba	p o~ b ª
porco	p o r k u
porta	p O r t ª
portugal	p u r t u g a l
praia	p r a i ª
pranchas	p r ª~ S ª S
prato	p r a t u
princesa	p r i~ s e z ª
pulseira	p u l s ª i r ª
páscoa	p a S k u ª
pássaro	p a s ª r u
pão	p ª~ u~
pé	p E
pés	p E S
pêra	p e r ª
pêssego	p e s @ g u
pêssegos	p e s @ g u S

pónei	p O n ª i
quadrado	k u ª d r a d u
quadro	k u a d r u
queijo	k ª i Z u
quente	k e~ t @
quivi	k i v i
rainha	R ª i J ª
raposa	R ª p o z ª
raquete	R ª k E t @
rato	R a t u
recta	R E k t ª
rectângulo	R E k t ª~ g u l u
rede	R e d @
regador	R @ g ª d o r
rei	R ª i
relógio	R @ l O Z i u
rena	R e n ª
renas	R e n ª S
retrato	R @ t r a t u
rinoceronte	R i n O s @ r o~ t @
robô	R o b o
rocha	R O S ª
rolha	R o L ª
roupa	R o p ª
rádio	R a d i u
rã	R ª~
régua	R E g u ª
sanita	s ª n i t ª
sapato	s ª p a t u
sapo	s a p u
secador	s @ k ª d o r
senhor	s @ J o r
sentada	s e~ t a d ª
sereia	s @ r ª i ª
seta	s E t ª
símbolo	s i~ b o l u
sinal	s i n a l
sino	s i n u
sofá	s u f a
sol	s O l
sumo	s u m u
tacho	t a S u
talher	t ª L E r
talheres	t ª L E r @ S
tambor	t ª~ b o r
tampa	t ª~ p ª
tangerina	t ª~ Z @ r i n ª
tartaruga	t ª r t ª r u g ª
telefone	t @ l @ f O n @
telemóvel	t E l E m O v E l
televisão	t @ l @ v i z ª~ u~
tesoura	t @ z o r ª
tigre	t i g r @
tocar	t u k a r
tomate	t u m a t @
triângulo	t r i ª~ g u l u
unha	u J ª

```
urso            u r s u
uva             u v ª
uvas            u v ª S
vaca            v a k ª
varanda         v ª r ª~ d ª
vaso            v a z u
vassoura        v ª s o r ª
veado           v i a d u
vela            v E l ª
velhinho        v E L i J u
velho           v E L u
velhote         v E L O t @
vestido         v @ S t i d u
vinho           v i J u
viola           v i O l ª
visão           v i z ª~ u~
xilofone        S i l O f O n @
zebra           z E b r ª
água            a g u ª
áriel           a r i E l
árvore          a r v u r @
índio           i~ d i u
íris            i r i S
óculos          O k u l u S
```

7.3. Anexo C - Scripts em DOS/BATCH de Treino dos Modelos

treino.bat

```
REM cd para directoria de treino

HLEd -l * -C configbasico.txt -d dic_newword.txt -i mlf\cut\mono_phones.mlf mkphones1.led
mlf\cut\MLF_cut2.mlf

REM set cdir=modelos\hmm0
REM - criar proto manualmente
REM list_train.txt criada com mfclist.m
REM - HCompV cria vFloors e actualiza proto
HCompV -C configbasico.txt -f 0.01 -S mlf/cut/MLF_cut2_scp_train.scp -m -M modelos/hmm0 proto_
REM - cuidado: o nome em ~h (neste caso "proto_") tem ser o mesmo do ficheiro (HCompV cracha)


REM criacao do MLF de treino com phonemlf.m
REM NOTA: os ficheiros MFC foram cortados para não terem <sil>,<noise>, etc. Criado MLF em
conformidade mantendo a ref do tempo inicial de cada label.
REM ver ficheiro: cutmlf.m


REM - fazer modelos\hmm0\macros  com opçoes e vFloors
echo macros - vFloors %%i - !date!  !time!
echo.
echo ~o >modelos\hmm0\macros
echo ^<STREAMINFO^> 1 39 >>modelos\hmm0\macros
echo ^<VECSIZE^> 39^<NULLD^>^<MFCC_0_D_A^>^<DIAGC^> >>modelos\hmm0\macros
type modelos\hmm0\vFloors >>modelos\hmm0\macros


REM - fazer hmmdefs = phones_GV.mmf -> clonagem dos modelos: ver protoscpclone.m
REM palavras
```

```
HERest -C configbasico.txt -T 7 -I mlf\cut\monophone_notime.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm0\macros -H modelos\hmm0\phones_GV.mmf -M
modelos\hmm1 phone_list_GV.txt
REM set cdir=modelos\hmm1
HERest -C configbasico.txt -T 7 -I mlf\cut\monophone_notime.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm1\macros -H modelos\hmm1\phones_GV.mmf -M
modelos\hmm2 phone_list_GV.txt
REM set cdir=modelos\hmm2
HERest -C configbasico.txt -T 7 -I mlf\cut\monophone_notime.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm2\macros -H modelos\hmm2\phones_GV.mmf -M
modelos\hmm3 phone_list_GV.txt
REM set cdir=modelos\hmm3
HERest -C configbasico.txt -T 7 -I mlf\cut\monophone_notime.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm3\macros -H modelos\hmm3\phones_GV.mmf -M
modelos\hmm4 phone_list_GV.txt
REM set cdir=modelos\hmm4
HERest -C configbasico.txt -T 7 -I mlf\cut\monophone_notime.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm4\macros -H modelos\hmm4\phones_GV.mmf -M
modelos\hmm5 phone_list_GV.txt
REM set cdir=modelos\hmm5
HERest -C configbasico.txt -T 7 -I mlf\cut\monophone_notime.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm2\macros -H modelos\hmm5\phones_GV.mmf -M
modelos\hmm6 phone_list_GV.txt
REM set cdir=modelos\hmm6
HERest -C configbasico.txt -T 7 -I mlf\cut\monophone_notime.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm2\macros -H modelos\hmm6\phones_GV.mmf -M
modelos\hmm7 phone_list_GV.txt

REM - Realinhar phones
HVite -l * -C configbasico.txt -o SW -H modelos\hmm7\macros -H modelos\hmm7\phones_GV.mmf -i
mlf\cut\aligned_phone.mlf -m -t 250.0 150.0 1000.0 -y lab -a -I mlf\cut\MLF_cut2.mlf -S
mlf\cut\MLF_cut2_scp_train.scp dic_newword.txt phone_list_GV.txt > logs\HVite_align_phone.log

REM - mais 2x HERest com alinhado
HERest -C configbasico.txt -T 7 -I mlf\cut\aligned_phone.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm7\macros -H modelos\hmm7\phones_GV.mmf -M
modelos\hmm8 phone_list_GV.txt
HERest -C configbasico.txt -T 7 -I mlf\cut\aligned_phone.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm8\macros -H modelos\hmm8\phones_GV.mmf -M
modelos\hmm9 phone_list_GV.txt
```

---

gmixture.bat

```
@ECHO OFF

set init=hmm9
for /L %%i in (2,1,16) do (


        md modelos\hmm_mix%%i_0
        md modelos\hmm_mix%%i_1
        md modelos\hmm_mix%%i_2
        md modelos\hmm_mix%%i_3
        REM apenas uma questão de um nome mais explícito:
        REM copy phone_list3.txt phone_list_notag.txt

        REM spliting de uma para 2 componentes Gaussianas:
        HHEd -H modelos\%init%\phones_GV.mmf -H modelos\%init%\macros -M modelos\hmm_mix%%i_0
hed\mix%%i.hed phone_list_GV > logs\logHHed_mix%%i.txt

        REM 3 passos de reestimação embebida:
        HERest -C configbasico.txt -T 7 -I mlf\cut\aligned_phone16.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm_mix%%i_0\phones_GV.mmf -H
modelos\hmm_mix%%i_0\macros -M modelos\hmm_mix%%i_1 phone_list_GV.txt

        HERest -C configbasico.txt -T 7 -I mlf\cut\aligned_phone16.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm_mix%%i_1\phones_GV.mmf -H
modelos\hmm_mix%%i_1\macros -M modelos\hmm_mix%%i_2 phone_list_GV.txt

        HERest -C configbasico.txt -T 7 -I mlf\cut\aligned_phone16.mlf -t 250.0 150.0 1000.0 -S
mlf\cut\MLF_cut2_scp_train.scp -H modelos\hmm_mix%%i_2\phones_GV.mmf -H
modelos\hmm_mix%%i_2\macros -M modelos\hmm_mix%%i_3 phone_list_GV.txt

```

```
        set init = hmm_mix%%i_3
)

@ECHO ON
```

---

teste.bat

```
REM cd para directoria de teste

REM creates de grammar network from the file taskgram generated with writehtkGrammar.m
HParse -C configbasico.txt ..\gram\gram ..\gram\net

HVite -T 01 -C configbasico.txt -H ..\test\modelos\hmm_mix16_3\macros -
H ..\test\modelos\hmm_mix16_3\phones_GV.mmf -S mlf\cut\MLF_cut2_scp_test.scp -l * -i recout.mlf -
w ..\gram\net.txt dic_lix.txt phone_list_GV.txt > log\HVite_rec.log

HResults -T 07 -e ??? lix -t -I MLF_cut2_lix.mlf phone_list_GV.txt recout.mlf > log\HRes.log
```

7.4. Anexo D – Scripts e Funções do Matlab

filtra_db.m

```
function filta_bd(listfile,dir_orig,dir_dest)
%
%Filtra BD passa-alto com freq. de corte 100Hz
%
%filtabd('list.txt','wavs','wav_hp')

if dir_orig(end)~='\', dir_orig=[dir_orig,'\']; end
if dir_dest(end)~='\', dir_dest=[dir_dest,'\']; end

list = textread(listfile,'%s');
N=length(list);
fs=8000;
f0=100;
wcn = (2*pi*f0/fs)/pi;
b = fir1(256,wcn,'high');
[H,w]=freqz(b,1);
plot(w/pi*fs/2,db(H)), grid

for n=1:N
    x=wavread([dir_orig,list{n}]);

    y = conv(x,b);
    maxy=max(abs(y));
    if maxy>0.99
        y = y./(maxy+0.0001);
    end
    y(1:128)=[];
    y(end-127:end)=[];

    wavwrite(y,fs,16,[dir_dest,list{n}]);
end
```

cleannoisemlf.m

```
function cleannoisemlf(in_mlf,out_mlf)
%function cleannoisemlf(in_mlf,out_mlf)
%
%Takes an mlf file and substitues the labels that end
% wtih $  and # by <noise>.

[pats,labs] = mlfRead2(in_mlf,'%f%f%s',char(9)); % del = char(9)
pats2 = pats;
labs2 = labs;
N = length(pats);

for n=1:N,
    for m=1:length(labs(n).lab)
        s = labs(n).lab{m};
        if s(end) == '#' || s(end) == '$'
            labs2(n).lab{m} = '<noise>';
        end
    end
end

mlfWrite2(pats2,labs2,out_mlf);

end
```

emptynoisemlf.m

```
function emptynoisemlf(mlf_file, out_mlf,delimiter)
%function emptnoiseymlf(mlf_file file_name,delimiter)
%Verifies if any of the pats of the mlf_file no spoken labs
% and only tags: <noise>, <sil>, <resp>, ...

fe = fopen(out_mlf,'wb');
empty = '';
if nargin<3, delimiter=char(9); end
[pats,labs] = mlfRead2(mlf_file,'%f%f%s',delimiter);

N = length(pats);
numpats = 0;

f = fopen(out_mlf, 'wb');
fprintf(f,'#!MLF!#\n');

for n=1 : N  %corre os ficheiros
    cont = 0;
    M = length(labs(n).lab);
    for m=1:M
        s = labs(n).lab{m};
        if  s(1) ~= '<' % spoken
            cont = cont + 1;
        end
    end
    if cont ~= 0
        fprintf(f, '"%s"\n',pats{n});
        for m=1:M
            fprintf(f, '%d\t%d\t%s\n',labs(n).ti(m),labs(n).tf(m),labs(n).lab{m});
        end
        fprintf(f, '.\n');
        numpats = numpats + 1;
    end

end

numpats
fclose(f);
```

```
end
```

---

formatmlf.m

```
function formatmlf(mlf_file,dir_dest,delimiter)
%function formatmlf(listfile,mlf_file,dir_dest)
% limpa as anotações de wav sem nada (lixo)
%cria lista de palavras e conta ocorrências
%cria lista de tags

MLFfile = mlf_file;
%list = textread(listfile,'%s');
if nargin<2, dir_dest=pwd; end
if nargin<3, delimiter=char(9); end
[pats,labs] = mlfRead2(MLFfile,'%f%f%s',delimiter);

N = length(pats)
words = ''; %locuções
tags = ''; %tags
flixo = ''; %pats de lixo
ilixo = []; %indices de lixo
nwords = 0; %indice locuções
ntags = 0; %indice tags
tagc = []; %num tags
wordc = []; %num locuções
fpats = ''; %pats
npats = ''; %pats noise

pl = strcat(dir_dest,'list_clean.txt');
pw = strcat(dir_dest, 'new_word.txt');
pt = strcat(dir_dest, 'new_unique_tags.txt');
pg = strcat(dir_dest, 'new_lixo.txt');
pn = strcat(dir_dest, 'new_noise.txt');
fl = fopen(pl,'wb');
fw = fopen(pw,'wb');
ft = fopen(pt,'wb');
fg = fopen(pg,'wb');
fn = fopen(pn,'wb');

cont = 1;
for n=1 : N  %corre os ficheiros
    M = length(labs(n).lab);
    nome = pats{n};
        name = horzcat(nome(3:(end-4)), '.wav');
        fpats = strvcat(fpats, name);
    for m=1 : M   %corre as anotações para cada ficheiro
        s = labs(n).lab{m};   %guarda o que está escrito

        if (M==1) && (strcmp(s,'lixo')) % não tem nada
            flixo = strvcat(flixo, nome); % guarda as ocorrências só com 'lixo'
            ilixo(cont) = n;
            cont = cont + 1;
        else  %guarda esta locução!


            if s(1) == '<' % é uma tag
                i = strmatch(s, tags,'exact');%verifica se já existe a tag
                if isempty(i) %se não existir : isempty=true
                    tags = strvcat(tags, s); %acrescenta à lista de tags
                    ntags = ntags + 1; %total tags
                    tagc(ntags) = 1;
                else %já existe a tag : isempty=false
                    tagc(i) = tagc(i)+1; %incrementa ocorrências
                end
            else % é uma palavra
                if s(end) == '$' % é para desprezar
                    labs(n).lab{m} = '<noise>';
                    fprintf(fn, '%s\t%s\n', s, nome);
                else
                    i = strmatch(s,words,'exact');%verifica se já existe a palavra
                    if isempty(i) %se não existir : isempty=true
                        fprintf(fw, '%s\n', s);
                        words = strvcat(words, s);
```

```
                        nwords = nwords +1; % total words
                        wordc(nwords) = 1;
                    else
                        wordc(i) = wordc(i)+1;
                    end
                end
            end
        end
    end
end
pats(ilixo) = []; %elimina locuções só com lixo
labs(ilixo) = [];
new_mlf = strcat(dir_dest,'clean_mlfdb.mlf');
mlfWrite2(pats, labs, new_mlf);
tags = sortrows(tags);
flixo = sortrows(flixo);

fclose(fw);
pw2 = strrep(pw,'_word','word');
f = fopen(pw2,'wb');
wlist = textread(pw,'%s','delimiter',char(9));
wlist = sortrows(lower(wlist));
wlist = unique(wlist);
for n = 1 : length(wlist)
    fprintf(f, '%s\n', wlist{n});
end
fclose(f);

for row = 1:size(pats,1)
    fprintf(fl, '%s\n', fpats(row,1:end));
end
for row = 1:size(tags,1)
    fprintf(ft, '%s\n', tags(row,1:end));
end
for row = 1:size(flixo,1)
    fprintf(fg, '%s\n', flixo(row,1:end));
end

fclose(fn);
fclose(fl);
fclose(ft);
fclose(fg);
```

---

cutmlf.m

```
function cutmlf (mlf_file,mfc_dir,dest_dir)
% Cuts mlf files to get only the  relevant info (cuts noise/breath/...)
% Separates mfl file into 2 files
%   - mlf with only the words
%   - mlf with remainder tags
% Uses two separate directories for the utterances and the remainder:
%       - .\cut;
%       - .\rest;
%
% the new /mlf files has an aditional fourth parameter with
% time information of the labels that refers to their original initial time
% in the original mlf file
% EXAMPLE:
%   ti  tf  lab     #original_ti
%   12  17  <sil>   #40

%% cutmlf('clean_mlfdb_v2.mlf','..\mfc')

MLFfile = mlf_file; %'clean_mlfdb_v2.mlf'
[pats,labs] = mlfRead2(MLFfile,'%f%f%s',char(9)); % del = char(9)
N = length(pats);

mkdir(dest_dir,'cut');
mkdir(dest_dir,'rest');
MLF_cut = strcat(dest_dir,'\cut\MLF_cut.mlf');
MLF_cut2 = strcat(dest_dir,'\cut\MLF_cut2.mlf');
MLF_rest = strcat(dest_dir,'\rest\MLF_rest.mlf');
MLF_rest2 = strcat(dest_dir,'\rest\MLF_rest2.mlf');
```

```
29
30      fec = fopen('MLF_cut_empty.txt', 'wb');
31      fer = fopen('MLF_rest_empty.txt', 'wb');
32      fc = fopen(MLF_cut, 'wb');
33      fc2 = fopen(MLF_cut2, 'wb');
34      fr = fopen(MLF_rest, 'wb');
35      fr2 = fopen(MLF_rest2, 'wb');
36
37      fprintf(fc,'#!MLF!#\n');
38      fprintf(fc2,'#!MLF!#\n');
39      fprintf(fr,'#!MLF!#\n');
40      fprintf(fr2,'#!MLF!#\n');
41
42      for n=1 : N
43      %disp(n);
44          M = length(labs(n).lab);
45          %nome = strrep(pats{n},'*',mfc_dir);    % ex: mfc_dir = '..\mfc'
46          nome = strrep(pats{n},'/','\');
47          nome(end-3:end) = '.mfc';
48          nome_o = strrep(nome,'*',mfc_dir);
49
50          nome_c = strrep(nome,'*',[mfc_dir, '\cut_clean']);
51          nome_r = strrep(nome,'*',[mfc_dir, '\rest_clean']);
52
53          [X,nSamples,sampPeriod,sampSize,parmKind]=readHTK(nome_o,0);
54          X_r=[];
55          X_c=[];
56          %fmfc_c = fopen(nome_c, 'wb');
57          %fmfc_r = fopen(nome_r, 'wb');
58
59          fprintf(fc, '"%s"\n',pats{n}); %ESTÁ AQUI '*/' A MAIS!!!!
60          fprintf(fc2, '"%s"\n',pats{n});
61          fprintf(fr, '"%s"\n',pats{n}); %ESTÁ AQUI '*/' A MAIS!!!!
62          fprintf(fr2, '"%s"\n',pats{n});
63
64          ti_r = 0;
65          ti_c = 0;
66
67          nSamples_c = 0;
68          nSamples_r = 0;
69          mt = 0;
70          mc = 0;
71          for m=1 : M   %corre as anotações para cada ficheiro
72                  lab = labs(n).lab{m};   %gets the comment
73                  ti = labs(n).ti(m);
74                  tf = labs(n).tf(m);
75                  d = tf - ti;  %duration
76
77                  %get correspondent sample number
78                  fi = floor(ti / sampPeriod) +1;
79                  ff = floor(tf / sampPeriod);
80                  if ff>size(X,2)
81                      fprintf('Tempo final errado em %s\n',pats{n});
82                      ff=size(X,2);
83                  end
84
85                  if lab(1) == '<' % it's a tag  -> MLF_rest
86                      mt = mt +1;
87                      fprintf(fr,'%.0f\t%.0f\t%s\t#%.0f\n',ti_r,ti_r + d,lab,ti);
88                      fprintf(fr2,'%.0f\t%.0f\t%s\n',ti_r,ti_r + d,lab);
89                      X_r = [X_r,X(:,fi:ff)];
90                      ti_r = ti_r + d;
91                      nSamples_r = nSamples_r + ff-fi + 1;
92                  else % it's an utterance -> MLF_cut
93                     mc = mc +1;
94                      fprintf(fc,'%.0f\t%.0f\t%s\t#%.0f\n',ti_c,ti_c + d,lab,ti);
95                      fprintf(fc2,'%.0f\t%.0f\t%s\n',ti_c,ti_c + d,lab);
96                      X_c = [X_c,X(:,fi:ff)];
97                      ti_c = ti_c + d;
98                      nSamples_c = nSamples_c + ff-fi + 1;
99                  end
100         end
101         if mt == 0
102             fprintf(fer,'%s\n',nome);
103         end
104         if mc == 0
105             fprintf(fec,'%s\n',nome);
106         end
107
```

```
    writeHTK(nome_c,X_c,nSamples_c,sampPeriod,sampSize,parmKind, 0);
    writeHTK(nome_r,X_r,nSamples_r,sampPeriod,sampSize,parmKind, 0);

    fprintf(fc, '.\n');
    fprintf(fr, '.\n');
    fprintf(fc2, '.\n');
    fprintf(fr2, '.\n');
end

fclose(fc);
fclose(fr);
fclose(fc2);
fclose(fr2);
fclose(fec);
fclose(fer);
```

---

mfclist.m

```
function mfclist(mlf_file)
%function mfclist(mlf_file)
%
%Given the directory of the .mlf file, generates 2 .scp files
% from the list of all pats in a mlf file.
%
%It generates a trainig list with 4/5 of the files and a
% test list with the remainder 1/5


nametrain = strrep(mlf_file,'.mlf','_scp_train.scp');
nametest = strrep(mlf_file,'.mlf','_scp_test.scp');

[pats,labs]=mlfRead2(mlf_file,'%f%f%s');
%files = dir (mfc_dir);
nfiles = '';

ftrain = fopen(nametrain,'wb');
ftest = fopen(nametest,'wb');
%N = length (files);
N = length (pats);
Ntrain = 4*round (N/5);
Ntest = N - Ntrain;

%list_train = strrep(fntransc,'.lab','.wav');

% for n=3:N  % start at n=3 to ignore .\ and ..\
%   scp_file = strrep(files(n).name,'S','..\mfc\cut\S');
%   fprintf(fmfc,'%s\n',scp_file);
% end


rand('seed',0);
i=randperm(N);
j=i(1:N);
for k=1:N
    scp_file = pats{j(k)};
    scp_file = strrep(scp_file,'*/S','../mfc/cut_clean/S');
    scp_file = strrep(scp_file,'.lab','.mfc');
    if k < Ntrain
        fprintf(ftrain,'%s\n',scp_file);
    else
        fprintf(ftest,'%s\n',scp_file);
    end
end;

fclose(ftrain);
fclose(ftest);
```

protoscpclone.m

```
function protoscpclone(in_proto,out_mmffile,dic)
%Clonagem
%ler proto em modelos\hmm0
%Escrever MMF com todos os modelos
[lab mod] = textread(dic,'%s%s','delimiter',char(9));

plist='';
N=length(mod);
for n=1:N
    c=textscan(mod{n},'%s');
    for m=1:length(c{1})
        cs=c{1}(m);
        i = strmatch(cs{1},plist,'exact');
        if isempty(i)
            plist=[plist,cs];
        end
    end
end

M=length(plist);
f=fopen('phonelist.txt','wb');
if f<3,error('mal'); end
for m=1:M
    fprintf(f,'%s\n',plist{m});
end
fclose(f);

f=fopen(in_proto,'rt');
if f<3, error('mal2'); end
t=char(fread(f)');
fclose(f);
i=strfind(t,'"proto"');
t1 =t(1:i);
t2 = t(i+7:end);

f=fopen(out_mmffile,'wb');
if f<3,error('mal4'); end
fprintf(f,'%s',t1(1:end-4));

for m=1:M
    name = plist{m};
    fprintf(f,'~h "%s"%s',name,t2);
end
fclose(f);
```

compareL1L2.m

```
function compareL1L2(ref_mlf,said_mlf,rec_mlf,dic,out_csv,mfc_dir)
%function compareL1L2(ref_mlf,said_mlf,rec_mlf,dic,out_csv,mfc_dir)
%
% ref_mlf   -> the reference mlf with what was asked of  the child
% said_mlf  -> mlf with what the child actually said
% rec_mlf   -> mlf output of the HVite recognition
%
% dic       -> dictionary
% mfc_dir   -> path to the mfc files : mfc_dir = '..\mfc\cut_clean\';
%
%Generates a CVS (TAB separated)file with a list of:
% REF SAID REC phones_Ref phones_free Nframes L1 L1-L2 ali_score al_sref al_sfree
%
%Ver. igual a compareL1L mas onde os tempos de início e fim de comando são
%tomados com base nos indicados em "MLF said_mlf"
% usage ex:
%compareL1L2('MLF_swlix.mlf','MLF_cut2_lix.mlf','recout.mlf','dic_newword.txt','swsaid.csv','..\m
fc\cut_clean\')
```

```
%compareL1L2('MLF_cut2_lix.mlf','MLF_cut2_lix.mlf','recout.mlf','dic_newword.txt','eqsaid.csv','.
.\mfc\cut_clean\')
%NOTA: o rec_mlf só serve para ir buscar os 'pats' de interesse (test)
%NOTA: assume que existe a mesma ordem dos pats e labs em SAID e REF.
%
%Serve para gerar out_csv para fazer histogramas de hipóteses H0/H1 com
%L1-L2, etc.

%cd C:\Users\Angel\Desktop\Thesis\childrenSR\DB;
[patref,labsref]=mlfRead2(ref_mlf,'%f%f%s');
[patsaid,labssaid]=mlfRead2(said_mlf,'%f%f%s');
[patrec,labsrec]=mlfRead2(rec_mlf,'%f%f%s%f');
Nrec = length(patrec);
%Ler dic (só deve ter as palavras-comando (sem [lix])
[words,transcs]=textread(dic,'%s%s','delimiter',char(9));

f = fopen(out_csv,'wb');
fprintf(f,'REF\tSAID\tREC\tphones\tFreephones\tNframes\tL1\tL1-
L2\talign_score\trefph_algn\tfreeph_algnt\n');
%toma todos os ficheiros MFC (áudio) em rec_mlf
for nrec=1 : Nrec
    Mrec = length(labsrec(nrec).ti);
    %no rec_mlf o comando reconhecido é sempre o último.
    labrec = labsrec(nrec).lab{Mrec};
    pat = strrep(patrec{nrec},'.rec','.lab');

    %finds same pat in REF
    iref = strmatch(pat,patref,'exact');
    m = strcmp('lix',labsref(iref).lab);
    m = find(m==0);
    m=m(end);
    labref=labsref(iref).lab{m};

%   sref = labsref(iref).lab{m};
%   %finds same pat in SAID
%   isaid = strmatch(wrec,patsaid,'exact');
%   m = 1;
%   while ~isempty(strmatch(labssaid(isaid).lab{m},'lix','exact'))
%       m = m + 1;
%   end

    %NOTA: assume que existe a mesma ordem em SAID e REF dos pats e labs
    isaid = strmatch(pat,patsaid,'exact');
    labsaid = labssaid(isaid).lab{m};

    ti = labssaid(isaid).ti(m);
    tf = labssaid(isaid).tf(m);

    f1 = fix(ti*1e-5)+1; %frame rate de 100 frames por segundo
    f2 = fix(tf*1e-5)-1;
    Nframes = (f2-f1)+1;

    % mfc_dir = '..\mfc\cut_clean\';
    mfc = strrep(patrec{nrec},'*/',mfc_dir);
    mfc = strrep(mfc,'.rec','.mfc');

    [X,nSamples,sampPeriod,sampSize,parmKind]=readHTK(mfc,0);
    writeHTK('temp.mfc',X(:,f1:f2),Nframes,sampPeriod,sampSize,parmKind,0);

    %Corre HVite com temp.mfc para conhecer a verosimilhança L2 (com gramática fones livre
(sequência qualquer de qualquer fone + GV)

    !HVite -C configbasico.txt -H hmm_mix16_3\phones_GV.mmf -l * -i tempfree.rec -w
phoneNet_GV.txt dic_phone_GV.txt phone_list_GV.txt temp.mfc

    %Corre HVite com uma gramática que é a sequência de fones de REF.
    %Para isso, vai buscar labref ao dicionário de transcrição dos comandos)
    %e gera uma REDE com HParse.

    %finds the labs.lab in the dictionary
    i = strmatch(labref,words,'exact');
    if isempty(i)
        error('não há');
    end
    i=i(1);
    % writes SAMPA of the word to make the grammar
    fg=fopen('gtemp.txt','wb');
    fprintf(fg,'(%s)\n',transcs{i});
    fclose(fg);
```

```
    phoneref = transcs{i};
    phoneref(phoneref==' ')=[];

    % makes wordnet from the grammar
    !HParse -C configbasico.txt gtemp.txt ntemp.txt
    !HVite -T 1 -C configbasico.txt -H hmm_mix16_3\phones_GV.mmf -l * -i tempcomm.rec -w
ntemp.txt dic_phone_GV.txt phone_list_GV.txt temp.mfc > log\HVite_2.log
    flog = fopen('log\HVite_2.log','rt');
    A='';
    while 1
        tline = fgetl(flog);
        if ~ischar(tline), break, end
        A=tline;
    end
    fclose(flog);
    if strcmp(A,'No tokens survived to final node of network')
        %Nframes too small to generate .rec file: put L1 =-infinity
        L1 = -1e10;
    else
        [pat1,labs1]=mlfRead2('tempcomm.rec','%f%f%s%f');
        %Tem a sequência de fones indicada em "ref_mlf" e em gtemp.txt/ntemp.txt
        L1 = sum(labs1.score)/Nframes; %soma dos likes dos fones normalizada pelo Nframes
    end

    [pat2,labs2]=mlfRead2('tempfree.rec','%f%f%s%f');
    L2 = sum(labs2.score)/Nframes;
    phonefree = '';
    for l=1 : length(labs2.lab)
        phonefree = [phonefree,labs2.lab{l}];
    end

    phonefree_uc = phonefree;
    phonefree_uc(find(diff(double(phonefree_uc))==0)+1)=[];
    phonefree_uc = SAMPA2SAMPAuc(phonefree_uc);

    phoneref_uc = phoneref;
    phoneref_uc(find(diff(double(phoneref_uc))==0)+1)=[];
    phoneref_uc = SAMPA2SAMPAuc(phoneref_uc);

    if length(phoneref) < 2 || length(phonefree) < 2
        score_ed = NaN;
        s1 = phoneref_uc; s2=phonefree_uc(1:length(phoneref_uc));
    else
        [score_ed,s1,s2] = editdistance(phoneref_uc,phonefree_uc);
    end

    fprintf(f,'%s\t%s\t%s\t%s\t%s\t%d\t%d\t%d\t%d\t%s\t%s\n',...
        labref,labsaid,labrec,phoneref,phonefree,Nframes,L1,L1-L2,score_ed, s1,s2);
end

fclose(f);
```

---

script para análise ROC e DET

```
% ROC and DET analysis


% errados
[wref1,ws1,rec1,ph1,phr1,nf1,L11,d1,s1,sa,sb]=textread('sw.csv','%s%s%s%s%s%d%f%f%d%s%s','delimit
er','\t','headerlines',1);
% correctos
[wref2,ws2,rec2,ph2,phr2,nf2,L12,d2,s2,sc,sd]=textread('eq.csv','%s%s%s%s%s%d%f%f%d%s%s','delimit
er','\t','headerlines',1);

% TP = sum (strcmp(ws1,rec1)); % true positive = Correctly accepted
% FP = sum(strcmp(wref2,rec2)); % false positive = Falsely accepted
% TN = sum(~strcmp(wref2,rec2)); % true negative = Correctly rejected
% FN = sum(~strcmp(ws1,rec1)); % false negative = Falsely rejected
%
% P = TP + FN;
% N = FP + TN;
% TPR = TP / P; % sensitivity
% FPR = FP / N;
% ACC = (TP + TN) / (P + N);
```

```
m=find(d1>100);
d1(m)=[];
Nbins=32;

%% L1-L2 correctos VS L1-L2 errados

% ROC
binmax=ceil(max([d1;d2]));
binmin=min([d1;d2]);
deltabin = (binmax-binmin)/Nbins;
bin = binmin + deltabin/2 +deltabin*(0:Nbins-1);
[count1] = hist(d1,bin); %errados
[count2] = hist(d2,bin); %correctos
plot(bin,count1,bin,count2,'r');
legend('L1-L2 Errados','L1-L2 Correctos');
grid on;

% DET
Ntrue = sum(count2);
Nfalse = sum(count1);
Ntotal = Ntrue + Nfalse;
fom = [d2;d1];
class0=false(size(fom));
class0(1:Ntrue)=true;
[NCA,NCR,NFA,NFR] = DET_DetectionErrorTradeoff(fom,class0);

%% align_score correctos VS align_score errados

% ROC
binmax2=ceil(max([s1;s2]));
binmin2=min([s1;s2]);
deltabin2 = (binmax2-binmin2)/Nbins;
bin2 = binmin2 + deltabin2/2 +deltabin2*(0:Nbins-1);
[count1] = hist(s1,bin2); %errados
[count2] = hist(s2,bin2); %correctos
plot(bin2,count1,bin2,count2,'r');
legend('Score de alinhanhemto - Errados','Score de alinhanhemto - Correctos');
grid on;

% DET
Ntrue = sum(count2);
Nfalse = sum(count1);
Ntotal = Ntrue + Nfalse;
fom = [s2;s1];
class0=false(size(fom));
class0(1:Ntrue)=true;
[NCA,NCR,NFA,NFR] = DET_DetectionErrorTradeoff(fom,class0);

%% Alguns valores de análise
avgL1 = mean(L11); % média L1 errados
avgd1 = mean(d1); % média d errados
avgL2 = mean(L12); % média L1 correctos
avgd2 = mean(d2); % média d correctos
medL1 = median(L11); % mediana L1 errados
medd1 = median(d1); % mediana d errados
medL2 = median(L12); % mediana L1 correctos
medd2 = median(d2); % mediana d correctos

[minL1,iminL1] = min(L11); % mínimo L1 errados
[mind1,imind1] = min(d1); % mínimo d errados
[minL2,iminL2] = min(L12); % mínimo L1 correctos
[mind2,imind2] = min(d2); % mínimo d correctos
[maxL1,imaxL1] = max(L11); %máximo L1 errados
[maxd1,imaxd1] = max(d1); %máximo d errados
[maxL2,imaxL2] = max(L12); %máximo L1 correctos
[maxd2,imaxd2] = max(d2); %máximo d correctos
```

8. Referências Bibliográficas

[1] A. Öster, D. House, A. Protopapas and A. Hatzis, "Presentation of a new EU project for speech therapy: OLP (Ortho-Logo-Paedia)," in *Proceedings Fonetik 2002*, 2002.

[2] M. Warschauer, "Technological Change and the Future of CALL," em *New Perspectives on CALL for Second and Foreign Language Classrooms*, Mahwah, NJ, Lawrence Erlbaum Associates, 2004, pp. 15-25.

[3] S. Bax, "CALL-Past, Present, and Future," *System*, vol. 31, n.º 1, pp. 13-28, 2003.

[4] A. Stoltzfus, "RosettaStone," Rosetta Stone (UK) Limited, [Online]. Available: http://www.rosettastone.co.uk/. [Acedido em 2012].

[5] "IBM Desktop ViaVoice," IBM, [Online]. Available: http://www-01.ibm.com/software/pervasive/viavoice.html. [Acedido em 2011].

[6] L. Heine, M. Witte, T. Diepstraten e T. Holl, "Babbel," Lesson Nine GmbH, 2008. [Online]. Available: http://pt.babbel.com/. [Acedido em 2011].

[7] "Babbel blog," Lesson Nine GmbH, 10 June 2010. [Online]. Available: http://blog.babbel.com/tech-background-babbel-speech-recognition/. [Acedido em 2012].

[8] "NUANCE," Nuance Communications, [Online]. Available: www.nuance.com. [Acedido em 2011].

[9] I. Lessner, "The Street," TheStreet, 02 10 2007. [Online]. Available: http://www.thestreet.com/s/nuance-buys-new-york-software-firm/newsanalysis/techsoftware/10382384.html?puc=_googlen?cm_ven=GOOGLEN&cm_cat=FREE&cm_ite=NA. [Accessed 2012].

[10] "LumenVox Speech Understood," LumenVox, [Online]. Available: http://www.lumenvox.com. [Acedido em 2012].

[11] "Accessibility - Windows 7 features - Microsoft Windows," Microsoft, [Online]. Available: http://windows.microsoft.com/en-US/windows7/products/features/accessibility.

[Acedido em 2012].

[12] N. Anderson, "Win 7's built-in speech recognition: a review | Ars Technica," ars technica, 1 June 2010. [Online]. Available: http://arstechnica.com/information-technology/2010/05/win-7s-built-in-speech-recognition-a-review/. [Acedido em 2012].

[13] G. Fant, Acoustic Theory of Speech Production, The Hague: Mouton & Co, 1960.

[14] "Vowel - Wikipedia, the free encyclopedia," Wikipedia, [Online]. Available: http://en.wikipedia.org/wiki/Vowel. [Accessed 2012].

[15] S. Ghai e R. Sinha, "Exploring the Effect of Differences in the Acoustic Correlates of Adults' and Children's Speech in the Context of Automatic Speech Recognition," *EURASIP Journal on Audio, Speech, and Music Processing,* vol. 2010, 2010.

[16] M. Gerosa, D. Giuliani e F. Brugnara, "Acoustic variability and automatic recognition of children's speech," *Speech Communication,* vol. 49, n.º 10-11, pp. 847-860, 2007.

[17] S. Ghai e R. Sinha, "An Investigation into the Effect of Pitch Transformation on Children Speech Recognition," em *TENCON*, Hyderabad, 2008.

[18] A. Potamianos and S. Narayanan, "A Review of the Acoustic and Linguistic Properties," in *Proc. Intern. Workshop on Multimedia Signal Processing*, Chania, Crete,Greece, 2007.

[19] A. Hagen, B. Pellom and R. Cole, "Children'S Speech Recognition With Application to Interactive Books and Tutors," in *Automatic Speech Recognition and Understanding*, 2003.

[20] R. D. Kent, "Anatomical and neuromuscular maturation of the speech," *Journal of Speech and*, vol. 9, p. 421–447, 1976.

[21] P. Martland, S. P. Whiteside, S. W. Beet and L. Baghai-Ravary, "Estimating Child and Adolescent Formant Frequency Values From Adult Data," in *Internat. Conf. Speech Language Processing*, Philadelphia, PA, 1996.

[22] A. Potamianos and S. Narayanan, "Robust Recognition of Children's Speech," *IEEE Transactions on Speech and Audio Processing,* vol. 11, no. 6, pp. 603-616, 2003.

[23] S. Narayanan e A. Potamianos, "Creating conversational interfaces for children," *IEEE Transactions on Speech and Audio Processing*, vol. 10, n.º 2, pp. 65-78, 2002.

[24] D. Elenius and M. Blomberg, "Comparing speech recognition for adults and children," in *FONETIK*, Stockholm, 2004.

[25] M. Gerosa, D. Giuliani, S. Narayanan e A. Potamianos, "A Review of ASR Technologies for Children's Speech," em *Proceedings of the Workshop on Child, Computer and Interaction*, Cambridge, MA, 2009.

[26] C. Lopes, A. Veiga and F. Perdigão, "A European Portuguese Children Speech Database for Computer Aided Speech Therapy," in *PROPOR 2012*, Coimbra, 2012.

[27] "fone - Infopédia," Porto Editora, [Online]. Available: URL: http://www.infopedia.pt/$fone. [Acedido em 2012].

[28] C. A. C. Lopes, "CLASSES FONÉTICAS ALARGADAS NO RECONHECIMENTO AUTOMÁTICO DE FONES," Universidade de Coimbra, Coimbra, 2011.

[29] H. P. Combrinck e E. C. Botha, "On The Mel-Scaled Cepstrum," em *Seventh Annual South African Workshop on Patter Recognition*, Pretoria, 1996.

[30] B. Bogert, M. Healy e J. Tukey, "The quefrency alanysis of time series for echoes: Cepstrum, Pseudo-Autocovariance, Cross-Cepstrum and Saphe Cracking," em *Proceedings of the Symposium on Time Series Analysis*, John Wiley & Sons, 1963, pp. 209-243.

[31] D. O'Shaughnessy, "Speech Communications: Human and Machine," in *Speech Communications: Human and Machine*, Addison-Wesley, 1987, p. 150.

[32] L. E. Baum e T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains," *The Annals of Mathematical Statistics*, vol. 37, n.º 6, pp. 1554-1563, 1966.

[33] S. E. Levinson, L. R. Rabiner e M. M. Sondhi, "Speaker Independent Isolated Digit Recognition Using Hidden Markov Models," em *International Conference on Acoustics, Speech, and Signal Processing*, Boston, 1983.

[34] L. R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,” em *Proceedings of the IEEE* , Orlando, 1989.

[35] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev e P. C. Woodland, The HTK Book. Revised for HTK version 3.4, Cambridge: Cambridge University Engineering Department, 2006.

[36] A. J. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *Information Theory, IEEE Transactions on,* vol. 13, n.º 2, pp. 260-269, 1967.

[37] G. D. Forney, “The Viterbi Algorithm,” *Proceedings of the IEEE,* vol. 61, n.º 3, pp. 268-278, 1973.

[38] M. S. Ryan e G. R. Nudd, “The Viterbi Algorithm,” University of Warwick Coventry, Coventry, 1993.

[39] L. R. Welch, “Hidden Markov Models and the Baum–Welch Algorithm,” *IEEE Information Theory Society Newsletter,* vol. 53, n.º 4, pp. 1,10-13, 2003.

[40] A. Veiga, S. Candeias and F. Perdigão, “Generating a Pronunciation Dictionary for European Portuguese Using a Joint-Sequence Model with Embedded Stress Assignment,” in *STIL,* Cuiabá, Mato Grosso, 2011.

[41] P. Boersma and D. Weenink, “Praat: doing phonetics by computer,” University of Amsterdam, Janeiro-Março 2010. [Online]. Available: http://www.fon.hum.uva.nl/praat/. [Accessed 2012].

[42] K. MacLean, “VoxForge,” Ken MacLean. [Online]. Available: http://www.voxforge.org/home. [Acedido em 2012].

[43] K. Sjölander, “An HMM-based system for automatic segmentation and alignment of speech,” em *Fonetik,* 2003.

[44] “Receiver operating characteristic - Wikipedia, the free encyclopedia,” Wikipedia, [Online]. Available: http://en.wikipedia.org/wiki/Receiver_operating_characteristic. [Acedido em 2012].

[45] A. Martin, G. Doddington, T. Kamm, M. Ordowski and M. Przybocki, "THE DET CURVE IN ASSESSMENT OF DETECTION TASK PERFORMANCE," in *Eurospeech*, Rhodes, Greece, 1997.

[46] L. d. C. Moutinho e R. M. Lima, "Desempenho Fonético em Crianças dos 3 a 7anos de idade no P.E.," em *III Congresso Internacional de Fonética e Fonologia, Universidade Federal Minas Gerais*, Minas Gerais, Brasil, 2007.

[47] S. Lee, A. Potamianos and S. Narayanan, "Acoustics of children's speech: Developmental changes of temporal and spectral parameters," *Journal of the Acoustical Society of America*, vol. 105, no. 3, 1999.

[48] K. Shobaki, J.-P. Hosom e R. A. Cole, "The OGI KIDS' Speech Corpus and Recognizers," em *International Conference on Spoken Language Processing (ICSLP)*, Beijin, 2000.

[49] A. Batliner, M. Blomberg, S. D'Arcy, D. Elenius, D. Giuliani, M. Gerosa, C. Hacker, M. Russell, S. Steidl and M. Wong, "The PF_STAR Children's Speech Corpus," in *Interspeech*, Lisboa, 2005.

[50] S. Young, "HTK," Cambridge University Engineering Department, [Online]. Available: http://htk.eng.cam.ac.uk/. [Acedido em 2011-2012].

[51] A. F. Martin and e. al., "The DET Curve in Assessment of Detection Task Performance," in *Eurospeech*, Rhodes, Greece, 1997.