ANÁLISES COMPARATIVAS EM SISTEMAS DE
RECONHECIMENTO DE VOZ

Amaro Azevedo de Lima

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Sergio Lima Netto, Ph.D.

Prof. Fernando Gil Vianna Resende Junior, Ph.D.

Prof. Márcio Nogueira de Souza, D.Sc.

Prof. Abraham Alcaim, Ph.D.

RIO DE JANEIRO, RJ - BRASIL
SETEMBRO DE 2000

LIMA, AMARO AZEVEDO DE

Análises comparativas em sistemas de reconhecimento de voz [Rio de Janeiro] 2000

IX,93 pp 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia Elétrica, 2000)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1.Reconhecimento de voz

2.Algoritmos de recorte

3.Ajuste temporal dinâmico

4.Modelos escondidos de Markov

5.Redes neurais artificiais

I.COPPE/UFRJ II.Título (série)

Dedicatória

Dedico esta tese:

a minha esposa, Ana Paula Gomes Mariano de Lima, que com sua paciência, dedicação e carinho, me deu forças e incentivo em todos os momentos.

Amaro Azevedo de Lima

Agradecimentos

Meus sinceros agradecimentos:

- aos meus pais, Benedito e Rita, pelo apoio e incentivo que sempre me deram;
- aos meus irmãos, Adherbal e Márcio, e cunhadas, Glorinha e Gislania, por todo auxílio que me deram e pela fundamental contribuição para minha formação;
- aos Professores Sergio Lima Netto e Fernando Gil Vianna Resende Junior, pela orientação e dedicação dispensadas durante todo o período de pesquisa;
- aos colegas Débora Andréa de Oliveira dos Santos , Ranniery da Silva Maia e Rodrigo Caiado de Lamare, por suas importantes colaborações;
- aos funcionários, alunos e professores do Laboratório de Processamento de Sinais da COPPE/UFRJ, por toda ajuda dispensada;
- CAPES, pela bolsa de estudos fornecida.

Amaro Azevedo de Lima

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

ANÁLISES COMPARATIVAS EM SISTEMAS DE RECONHECIMENTO DE VOZ

Amaro Azevedo de Lima

Setembro/2000

Orientadores: Sergio Lima Netto
Fernando Gil Vianna Resende Junior

Programa: Engenharia Elétrica

Este trabalho apresenta um estudo acerca dos sistemas de reconhecimento de palavras isoladas, em particular dígitos da língua portuguesa falada no Brasil.

São analisadas as performances de três algoritmos de recorte propostos na literatura, e como resultado desta análise é proposto um novo algoritmo que é também comparado aos demais.

São realizados ainda estudos dos diversos tipos de coeficientes (características) que podem ser extraídos do sinal de voz para efeito de reconhecimento. Este estudo inclui algoritmos recentemente propostos na literatura como o LPC-mel de Tokuda [1] e os PLP de Hermanski [2].

Em seguida, estudamos o que consideramos as três principais técnicas de classificadores para sistemas de reconhecimento: o ajuste temporal dinâmico (*dynamic time warping*, DTW), os modelos escondidos de Markov (*hidden Markov models*, HMM) e as redes neurais artificiais (*artificial neural networks*, ANN). As performances destas técnicas para reconhecimento são comparadas em termos de percentual de acerto e complexidade computational.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

COMPARATIVE ANALYSES IN SPEECH RECOGNITION SYSTEMS

Amaro Azevedo de Lima

September/2000

Advisors: Sergio Lima Netto

Fernando Gil Vianna Resende Junior

Department: Electrical Engineering

This work presents a thorough comparitive study on the performances of speech recognition systems for isolated words, in particular, the digits in Brazilian Portuguese.

We analyze the performances of three end-point detection algorithms found in the literature, and as a result of such analysis end up proposing a new algorithm, whose performance is also compared to the other three.

We also discuss the calculation of several speech characteristic parameters suitable for speech recognition systems. This study includes recently proposed algorithms such as the LPC-mel proposed by Tokuda [1] and the perceptual linear prediction (PLP) proposed by Hermansky [2].

We then compare the implementation of the three main techniques for speech classification: dynamic time warping (DTW), hidden Markov models (HMM), and artificial neural networks (ANN). Such techniques are compared in terms of their correct recognition rates and computational complexity when applied to speech recognition systems.

Sumário

Capítulo 1

Introdução

1.1 Introdução

A possibilidade de se falar sem restrições com a máquina tem fascinado os cientistas por décadas. Esta interação entre o homem e a máquina tem sido uma das áreas de maior interesse dos pesquisadores, pois certamente a funcionalidade de um sistema que permita esta comunicação promoveria uma revolução da informação, possibilitando a todos o acesso a sistemas que até então eram restritos a pessoas especializadas.

Após muitos anos de dedicação e pesquisa, esta interação entre o homem e a máquina tem se tornado realidade, de forma contínua e gradual. Atualmente, ainda não existe um sistema capaz de compreender tudo que o homem fala livremente (sem restrições). Um dos maiores problemas nesta comunicação é fazer com que, por exemplo, um computador compreenda o que é falado para ele, ou seja, fazer com que a máquina converta os sinais acústicos da fala em informação útil. Este problema trata do reconhecimento de sinais de fala que atualmente está ultrapassando a barreira da teoria e entrando na vida doméstica das pessoas, através de produtos com alto apelo comercial, como por exemplo, telefones móveis que atendem a comandos de voz.

Reconhecer sinais de fala envolve várias tecnologias e se torna cada vez mais um campo desafiador para os pesquisadores. De fato, por mais desenvolvidos que

estejam os sistemas de reconhecimento, ainda falta muito para eles atingirem o grau de liberdade com que as pessoas se comunicam.

1.2 Definindo o problema

Os sistemas de reconhecimento de sinais de fala possuem características específicas que determinam sua complexidade, obviamente em função da tarefa a ser realizada. Na Tabela 1.1 exemplificamos os parâmetros mais comuns que caracterizam os sistemas e suas aplicações.

Tabela 1.1: Parâmetros gerais que caracterizam a complexidade de sistemas de reconhecimento de fala.

Parâmetros	**Aplicações**		
modo de falar	palavras isoladas	×	fala contínua
estilo de falar	fala por leitura	×	espontânea
"envolvimento" do orador	dependente	×	independente do orador
vocabulário	pequeno	×	grande
perplexidade	pequeno (< 10)	×	grande (> 100)
SNR (razão sinal ruído)	alto (> 30 dB)	×	baixo (< 10 dB)
transdutor	microfone	×	telefone

Os sistemas de reconhecimento podem ser classificados em relação ao modo de fala como um sistema de palavras isoladas, palavras conectadas ou de fala contínua. Em um sistema de palavras isoladas, temos como entrada do sistema uma palavra de cada vez. Em geral, nos sistemas de palavras conectadas pronunciamos as palavras de forma natural somente restringindo o tamanho do vocabulário utilizado. E finalmente, no processamento da fala contínua a idéia é não se realizar restrição alguma quanto à pronúncia das palavras e ao tamanho do vocabulário.

Quanto ao estilo de fala, os sistemas são classificados de fala espontânea ou de leitura. A maior dificuldade se aplica a sistemas de fala espontânea, pois costuma-se

realizar contrações fonéticas entre palavras, enquanto que no processo de leitura as contrações fonéticas não ocorrem muito significativamente, em geral.

Os sistemas independentes do orador são aqueles treinados por vários oradores e que podem ser utilizados por outros oradores que não participaram desta etapa de treinamento. Naturalmente sistemas independentes do orador têm maior aplicabilidade do que sistemas dependentes do orador, em geral.

Com relação às tarefas a serem realizadas pelo sistema, podemos citar o tamanho do vocabulário que está relacionado ao número de palavras a serem reconhecidas e o modelo de linguagem que restrige as possíveis combinações de palavras de acordo com as regras gramaticais da língua utilizada.

A perplexidade é definida pela média geométrica do número de palavras que pode vir depois de outra quando aplicado o modelo da língua. Fatores externos também recebem igual importância como por exemplo, a influência do ruído ambiente e o posicionamento e tipo de transdutor utilizado para converter as ondas sonoras em sinais elétricos.

Como já foi descrito anteriormente, a complexidade do sistema depende de várias características. Além dessas possibilidades, existem variações associadas ao sinal que afetam bastante a tarefa de reconhecer um sinal de voz. Por exemplo, mudanças no ruído ambiente, no transdutor, no estado emocional do locutor ou mesmo na velocidade da fala podem se tornar problemas bem complexos para os sistemas de reconhecimento. Como se não bastasse, ainda há diferenças de formação sócio-lingüística, de dialeto e da forma e tamanho do aparelho articulador da fala.

Toda esta variabilidade associada à tarefa a ser executada define a complexidade dos sistemas de reconhecimento, e de acordo com estas características utilizam-se diferentes técnicas com o objetivo de otimizar o reconhecimento.

1.3 Sistema de reconhecimento genérico e simplificado

Nesta seção é apresentado o modelo genérico de um sistema de reconhecimento bastante simplificado, porém diretamente relacionado com os sistemas implementados neste trabalho. A descrição apresentada é tipica de um sistema de reconhecimento de palavras isoladas com um pequeno vocabulário. Basicamente, este sistema é dividido em três blocos: recorte, extração de parâmetros e reconhecimento, como é mostrado na Figura 1.1.

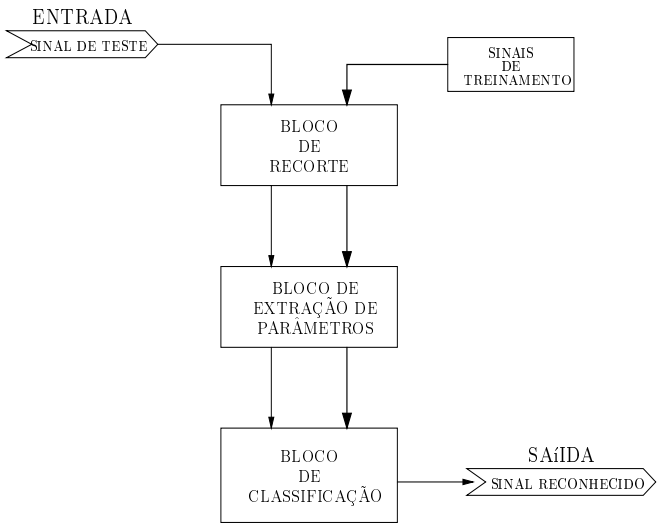

Figura 1.1: Sistema de reconhecimento genérico e simplificado.

O bloco de recorte trata da extração do sinal de fala, ou seja, a elocução é capturada e tratada de tal forma que só seja mantida a informação onde existe fala. O objetivo deste bloco é eliminar ruído ou informação indesejável que aparece antes ou depois do sinal de fala a ser analisado.

O bloco de extração de parâmetros tem como finalidade representar o sinal de fala de uma forma reduzida e que facilite a comparação entre segmentos ressaltando as diferenças entre sinais. Por exemplo, 20 ms de sinal de fala amostrado em 8 kHz geram uma seqüência de 160 pontos. Com uma extração criteriosa de parâmetros podemos reduzir este número para uma seqüência de 20 coeficientes por segmento

de 20 ms, por exemplo.

O bloco de reconhecimento pode ser subdividido em dois: treinamento e classificação. O treinamento se refere à etapa na qual o sistema é treinado para reconhecer os sinais desejados. Por exemplo, se desejamos que o sistema reconheça os dez dígitos ("0", "1", ..., "9") para a língua portuguesa falada no Brasil, será preciso um certo número de sinais para cada classe (dígito) para obter modelos ou sinais de referência. No caso de sistemas dependentes do locutor os sinais de treinamento são gravados por um único orador, e em sistemas independentes do locutor os sinais de treinamento são gravados por um grande número de oradores, preferivelmente de diferentes idade, sexo e região para que o sistema possa ser o mais abrangente possível.

A etapa de classificação é onde ocorre o processamento final do sistema. Este depende de todas as etapas anteriormente mencionadas para executar seu próprio processamento, gerando o resultado final do reconhecimento. Existem várias técnicas utilizadas para implementar o algoritmo de reconhecimento. Neste trabalho são analisados: o ajuste temporal dinâmico (*dynamic time warping* - DTW), os modelos escondidos de Markov (*hidden Markov models* - HMM) e as redes neurais artificiais (*artificial neural networks* - ANN).

1.4 Estrutura da tese

O objetivo desta tese é realizar um estudo abrangente sobre sistemas de reconhecimento de palavras isoladas, englobando as etapas referentes ao processamento, desde o recorte da palavra, passando pela extração de parâmetros até chegar na classificação. As técnicas utilizadas nos classificadores serão DTW, HMM e ANN.

A técnica DTW foi uma das primeiras a surgir e ser aplicada no reconhecimento de sinais de fala, sendo caracterizada pelo alinhamento temporal de dois sinais de fala gerando um resultado de similaridade entre eles. Esta técnica foi amplamente explorada na década de 1980 e seu custo computacional se torna bastante elevado ao ser aplicada a tarefas mais complexas como sistemas com grande vocabulário e

independentes do orador.

A segunda cronologicamente falando é a técnica HMM que se trata de um modelo estocástico, no qual a geração de cadeias de fonemas e/ou segmentos de fala são representados probabilisticamente através de processos de Markov. Esta técnica diminuiu consideravelmente o custo computacional de tarefas mais complexas realizadas pelo DTW, e tem sido o alvo das pesquisas mais recentes gerando ótimos resultados.

A terceira técnica analisada é ANN que surgiu com a esperança de solucionar o problema de se modelar um sistema que tivesse resposta similar a resposta do sistema auditivo humano. Porém, as ANN não têm se mostrado tão eficientes quanto esperado, atingindo no máximo resultados equivalentes aos resultados já obtidos com técnicas mais consolidadas. Esta técnica continua sendo pesquisada e até aplicada a sistemas híbridos em conjunto com o DTW ou HMM.

A organização desta tese se dá nos seguintes moldes: o Capítulo 2 descreve técnicas de recorte de palavras com o intuito de eliminar informação indesejável presente no sinal, ou seja, extrair por exemplo o ruído ambiente que aparece antes e depois da palavra pronunciada. Neste capítulo são descritos e implementados três algoritmos [3, 4, 5] de recorte e é proposto um novo algoritmo.

A próxima etapa desenvolvida no Capítulo 3 é a extração de parâmetros onde são apresentados os aspectos teóricos de vários coeficientes que podem ser extraídos de um sinal de voz. Neste capítulo são descritas as implementações de algoritmos de extração dos coeficientes LPC (*linear predictive coding*) [6, 3], cepestrais usando FFT [6, 3], cepestrais derivados dos LPC [6, 3], mel-cepestrais pela definição de Deller et al. [3], mel-cepestrais pela definição de Rabiner et al. [6], mel-cepestrais derivados dos LPC recursivamente [1] e os PLP (*perceptual linear predictive*) [2]. As análises realizadas neste capítulo são puramente teóricas, enquanto as análises práticas são apresentadas nos capítulos subseqüentes onde estes parâmetros são aplicados diretamente a sistemas de reconhecimento.

No Capítulo 4 é descrita técnica de reconhecimento relativa ao algoritmo DTW. São discutidos os aspectos teóricos e práticos do assunto, além de uma apre-

sentação completa do sistema implementado com esta técnica. Os resultados são mostrados para os quatro algoritmos de recorte desenvolvidos no Capítulo 2 e para todos os parâmetros desenvolvidos no Capítulo 3, com a restrição do sistema ser dependente do orador.

No Capítulo 5 são apresentados os aspectos teóricos e práticos do sistema implementado usando o classificador HMM. Os resultados deste capítulo são mostrados para os métodos de recorte vistos anteriormente e para três tipos de coeficientes: cepestrais, mel-cepestrais derivados do LPC e PLP.

No Capítulo 6 é estudado o classificador ANN, cujos testes realizados seguiram o mesmo padrão apresentado no Capítulo 5.

Finalmente a conclusão apresentada no Capítulo 7 faz uma análise das técnicas descritas na tese ressaltando os aspectos didáticos e atuais do trabalho, além de propor algumas possibilidades de continuidade a cada uma das etapas aqui desenvolvidas.

Capítulo 2

Comparação de técnicas de recorte

2.1 Introdução

A detecção de início e fim da elocução a ser processada é um problema fundamental com que sistemas de reconhecimento de voz se deparam. Uma detecção falha pode acarretar erro nas etapas subseqüentes de processamento, fazendo com que o classificador perca consideravelmente sua eficiência [6].

Os algoritmos de recorte podem ser separados em três tipos: explícito, implícito e híbrido [6]. Neste trabalho abordamos somente métodos explícitos de recorte, pois em geral os métodos implícitos e híbridos não são adequados a aplicações em tempo real devido à grande complexidade computacional. Além disso, métodos híbridos utilizam estimativas iniciais de recorte obtidas com métodos explícitos.

Neste capítulo são abordados três algoritmos bastante difundidos na literatura [7, 8, 5], assim como um novo algoritmo que esta sendo proposto, mostrando seus conceitos e implementações. Na Seção 2.2 são introduzidos alguns conceitos para iniciarmos o estudo dos métodos de recortes. O primeiro algoritmo desenvolvido por Rabiner e Sambur [7], utiliza medidas simples como energia e taxa de cruzamento por zero para analisar a presença de fala no sinal. O próprio Rabiner, juntamente com outros pesquisadores, propôs outro algoritmo [8] após constatar que o método anterior [7] não funcionava devidamente para baixas taxas de amostragem. Este segundo algoritmo utiliza somente a medida de energia, porém seu processamento é

bem mais elaborado que o primeiro, como será visto adiante. O terceiro algoritmo utiliza técnicas mais complexas que os dois primeiros, como por exemplo, o método de k-médias modificado [5, 9]. O quarto algoritmo é baseado na utilização de idéias descritas nos algoritmos anteriormente mencionados. Nas Seções 2.7 e 2.8 são apresentados técnicas complementares de recorte e resultados. Na conclusão do capítulo (Seção 2.9) são feitas comparações entre os algoritmos de recorte aqui discutidos.

2.2 Considerações iniciais

Um aspecto importante a se analisar nos algoritmos de recorte é a sua sensibilidade em relação a certas perturbações sonoras. As perturbações sonoras mais comuns ocorridas em sinais de fala são o "click" e o sopro. O "click" pode ocorrer devido ao estalar da língua ou dos lábios, quando ressecados, antes ou depois de se pronunciar a palavra desejada. Já o sopro ocorre devido a respiração mais forte, aparecendo em geral, após a palavra pronunciada.

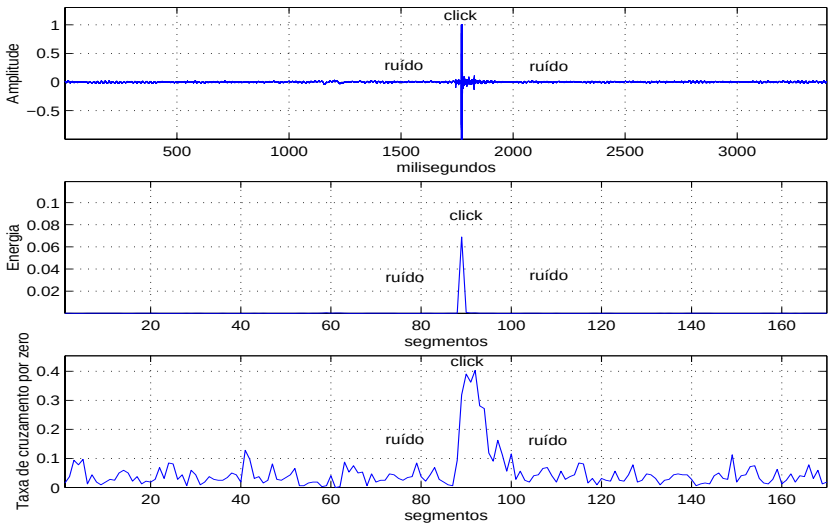

Figura 2.1: Análise do "click".

Nas Figuras 2.1 – 2.3 são realizadas algumas análises de tais perturbações. Os sinais representados nas figuras foram amostrados em 16 kHz, e sua segmentação obtida com janelas retangulares de 20 ms sem superposição. Nestas figuras são

apresentadas a forma de onda do sinal de voz, a energia e a taxa de cruzamento por zero em cada segmento do sinal testado.

Na Figura 2.1 é analisado o "click" comparativamente com o ruído de fundo. Tal perturbação, em geral, possui duração muito curta, neste exemplo aproximadamente 70 ms. Observa-se também que durante o "click" o nível de energia é alto, em relação ao ruído.

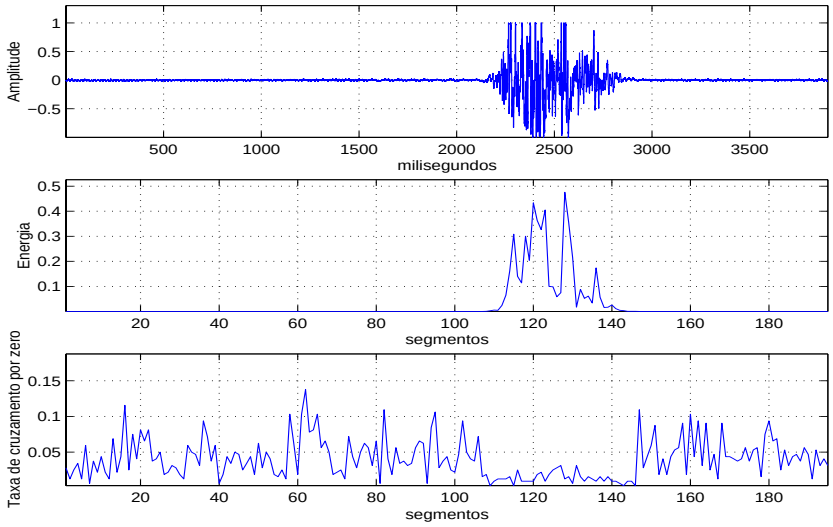

Figura 2.2: Análise do sopro.

Na Figura 2.2 é mostrado um exemplo do sopro, onde pode-se observar que, em geral, sua duração e seu nível de energia são relativamente maiores que os do "click", e sua taxa de cruzamento por zero é relativamente menor que a taxa do ruído.

Na Figura 2.3 a palavra pronunciada foi o dígito "oito", e durante a gravação foi inserido no sinal "click" e sopro, antes e após a palavra respectivamente. Pode-se observar que o "click" e o sopro possuem níveis de energia da mesma ordem do nível de energia palavra. É possível também distinguir a transição entre o fim do "click" e o ruído, e observar a presença do sopro pela taxa de cruzamento por zero.

A presença e separação do "click" é bastante clara com a simples observação da forma de onda do sinal, porém o sopro já não é tão evidente assim. A informação necessária para separar o sopro da palavra pode ser encontrada na taxa de

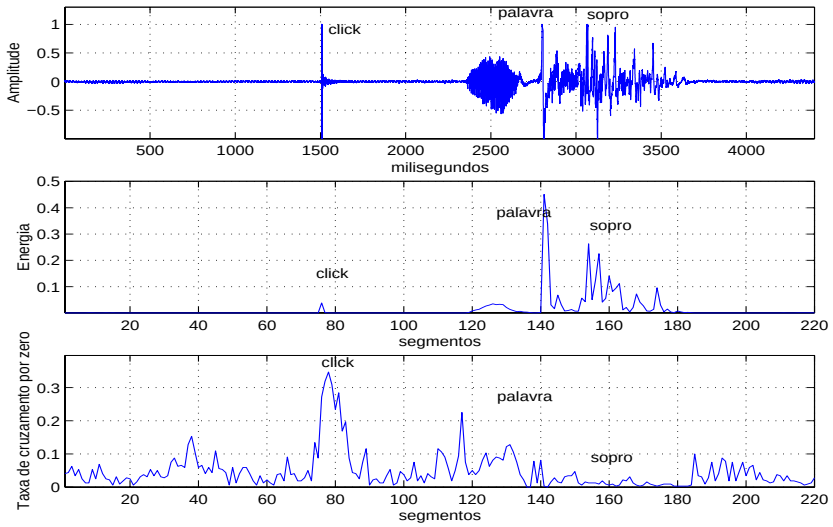

Figura 2.3: Análise de um sinal que apresenta: "click", palavra e sopro.

cruzamento por zero do sinal. Porém na maioria dos algoritmos de recorte esta informação é negligenciada pela presença de um alto nível de energia durante o sopro, de tal modo, dificultando os algoritmos de realizarem recortes capazes de suprimir este tipo de perturbação sonora. Em geral, os algoritmos de recorte realizam uma análise por energia estabelecendo uma região de recorte por energia e depois realizam uma análise da taxa de cruzamento por zero, fora da região recortada por energia. Como o "sopro" possui uma energia considerável em relação a palavra, o que pode ser visto na Figura 2.3, ele já se enquadra diretamente na região recortada por energia, não sendo nem analisada a sua taxa de cruzamento por zero.

2.3 Algoritmo 1

O objetivo do algoritmo de Rabiner e Sambur [7] é detectar onde se inicia e termina a informação de fala dentro do sinal, utilizando taxa de cruzamento por zero, energia e estatísticas tanto de primeira quanto de segunda ordem do sinal de voz e do ruído ambiente. Este algoritmo é um dos mais utilizados e simples de ser compreendido e implementado, podendo ser encontrado em [10, 11, 3]. A seguir serão definidas as medidas utilizadas na sua implementação.

2.3.1 Taxa de cruzamento por zero

A taxa de cruzamento por zero $ZC(m)$ de um segmento de sinal de fala é o número de vezes que o sinal cruza o eixo onde a amplitude do sinal $s(n)$ é zero, normalizado pelo tamanho do segmento:

$$ZC(m) = \frac{1}{N} \sum_{n=m-N+1}^{m} \frac{|sgn\{s(n)\} - sgn\{s(n-1)\}|}{2} \tag{2.1}$$

onde

$$sgn\{s(n)\} \quad = \quad \begin{cases} +1, & s(n) \geq 0 \\ -1, & s(n) < 0 \end{cases}$$

e N é o número de amostras de um segmento.

2.3.2 Energia

A energia de um segmento de sinal de voz é dada por

$$E(m) = \sum_{n=m-N+1}^{m} s^2(n) \tag{2.2}$$

2.3.3 Descrição do algoritmo 1

Inicialmente, o algoritmo de Rabiner e Sambur usa algumas estatísticas do ruído de fundo, que obrigatoriamente deve estar contido no início das amostras do sinal de fala, para definir três limiares que serão utilizados para realizar o recorte. Estes limiares são definidos por:

- Limiar alto de energia: $le1 = \mu_e + A\sigma_e^2$
- Limiar baixo de energia: $le2 = \mu_e + B\sigma_e^2$
- Limiar de cruzamento por zero: $lz = \mu_z + C\sigma_z^2$

onde μ_e é a média da energia do ruído, σ_e^2 é a variância da energia do ruído, μ_z é a média da taxa de cruzamento por zero do ruído e σ_z^2 é a variância da taxa de cruzamento por zero do ruído. Heuristicamente, estes valores são calculados em pelo menos 10 segmentos, e usam-se A = 10, B = 5 e C = 3 [10].

A primeira etapa do processamento consiste na segmentação do sinal no qual se deseja efetuar o recorte, em janelas com duração entre 10 e 20 ms. Para cada segmento m calculam-se os valores de energia $E(m)$ e a taxa de cruzamento por zero $ZC(m)$. A partir destes valores nos 10 primeiros segmentos são obtidos os limiares $le1$, $le2$ e lz, como descritos acima.

O limiar alto de energia $le1$ é usado para estimar de forma "grosseira" um possível extremo da palavra. Depois com o limiar baixo $le2$ encontram-se os possíveis extremos de palavra de forma um pouco mais "refinada". Por fim, com o limiar de cruzamentos por zero lz analisa-se a existência de segmentos surdos no ínicio e no fim do sinal. Estes limiares são obtidos experimentalmente por intermédio das estatísticas de primeira e segunda ordem da energia e da taxa de cruzamento por zero do ruído de fundo, que está sempre representado nos primeiros 200 ms dos sinais gravados (característica do sinal gravado).

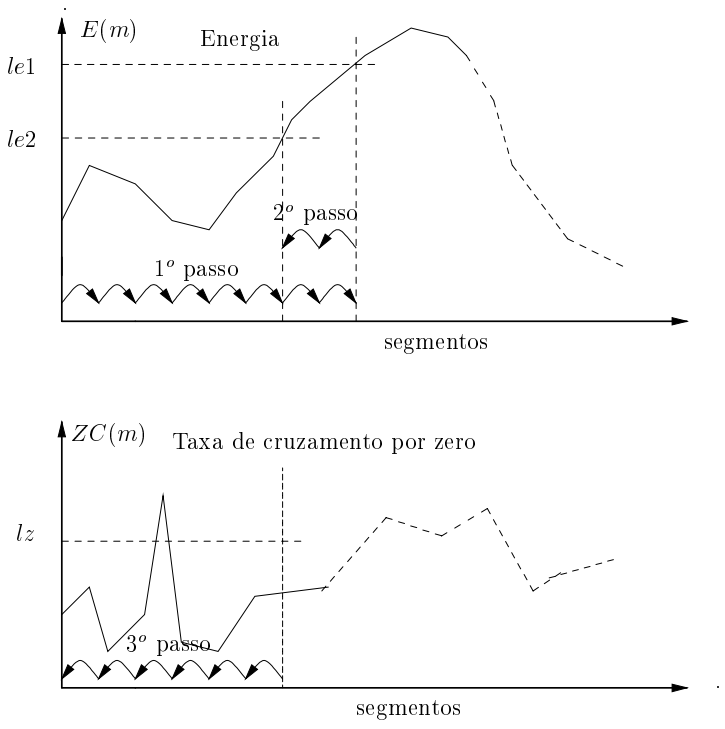

Figura 2.4: Passos executados durante o processamento do algoritmo de recorte de Rabiner e Sambur (somente o início do sinal).

A segunda etapa do processo de recorte pode ser subdividida em três passos. No passo 1, caminha-se em direção ao centro do sinal a partir dos extremos, como mostrado na Figura 2.4 (onde somente a primeira metade do sinal processado é mostrada), até encontrar os primeiros segmentos que tenham valor de energia maior ou igual a $le1$, marcando-se estes segmentos. Com esta marcação obtém-se uma primeira estimativa do recorte.

No passo 2, a partir dos segmentos marcados, caminha-se em direção aos extremos, até encontrar segmentos que tenham energia menor ou igual a $le2$, então marcam-se estes segmentos, que dão uma estimativa mais precisa que a anterior para o início e fim da palavra.

No passo 3, analisa-se a possibilidade de ocorrerem sons referentes a fonemas fricativos, bilabiais ou qualquer outro que seja caracterizado por baixa energia e alta taxa de cruzamento por zero. Então, a partir dos pontos obtidos no passo 2 caminha-se no máximo 25 segmentos em direção aos extremos observando a ocorrência de segmentos que tenham valor da taxa de cruzamento por zero superior a lz. Caso haja três ou mais segmentos que excedam este limiar, marca-se como ponto de recorte o último segmento analisado que exceda o limiar. Caso o número de ocorrências não seja maior ou igual a três, mantém-se o ponto de recorte obtido no passo 2.

O sinal de voz analisado poderá ser descartado se durante o passo 1 não for encontrada uma primeira estimativa do recorte.

Pode-se observar claramente os passos explicados anteriormente na Figura 2.4, ressaltando novamente o fato da figura representar o processamento somente da primeira metade do sinal a ser recortado.

2.3.4 Pseudo-código

A partir do sinal de voz original $s(n)$ com $n = 1, 2, \ldots, N_s$ e amostradas a uma freqüência Fs, obtêm-se as amostras de $s(n)$ referentes aos 100 ms iniciais do sinal que representam o ruído ambiente no momento da gravação. Estas amostras são segmentadas em janelas retangulares de tamanho N sem superposição. Através dos segmentos obtidos anteriormente, calculam-se os valores de μ_e, μ_z, σ_e e σ_z que

por conseguinte, possibilitará o cálculo dos limiares $le1$, $le2$ e lz.

Na Tabela 2.1, I_{le1}, F_{le1}, I_{le2}, F_{le2}, I_{lz} e F_{lz} representam respectivamente os segmentos inicial e final de recorte obtidos através dos limiares $le1$, $le2$ e lz.

Tabela 2.1: Pseudo-código do algoritmo 1.

1^a Etapa:		
	Segmentar $s(n)$.	
	Calcular $E(m)$ e $ZC(m)$, para $m = 1, 2, \ldots, L$.	
	Calcular $le1$, $le2$ e lz.	
2^a Etapa:		
	passo 1:	Obter I_{le1} e F_{le1} usando o limiar $le1$.
	passo 2:	Obter I_{le2} e F_{le2} usando o limiar $le2$.
	passo 3:	Obter I_{lz} e F_{lz} usando o limiar lz.

2.4 Algoritmo 2

O algoritmo 2 [8] de Rabiner et al. foi proposto quando se verificou que o algoritmo anterior se mostrava pouco eficiente para baixas taxas de amostragem, pois estas podem gerar um sinal cuja taxa de cruzamento por zero seja pouco acurada [8]. De fato, o algoritmo 2 utiliza somente a energia computada nos segmentos do sinal gravado, que não é afetada de forma significativa pela taxa de amostragem. Através de um processamento um pouco mais refinado que o anterior e com o auxílio do cálculo da energia nos segmentos, encontram-se pulsos de energia que são "ordenados" de modo a formar possíveis início e fim de recorte. Esta ordenação será decrescente em relação ao recorte mais provável, ou seja, a primeira dupla de segmentos que representam o início e fim da palavra serão os mais prováveis de estar realizando o recorte corretamente, e assim por diante.

Este algoritmo pode ser dividido em três etapas, porém antes de se iniciar o processamento é preciso inicializar o sistema para carregar algumas variáveis pertinentes ao processo. Estas variáveis são os limiares de energia e de tempo. Durante

a inicialização, calcula-se Q, que é dado pelo logaritmo da energia média do ruído de fundo.

O cálculo de Q pode ser realizado de várias formas, porém a referência em questão [8] nos apresenta dois métodos. No primeiro, o ruído é gravado durante a inicialização do sistema, antes de se executar qualquer processamento. Este método é equivalente ao apresentado no algoritmo anterior, onde são utilizados os primeiros segmentos do sinal de fala. Já no segundo, realiza-se uma análise do ruído de fundo a partir das amostras do sinal de mais baixas energias, assim podendo reajustar Q adaptativamente. Esta análise é feita através de um histograma de dez faixas que compreendem o intervalo entre o valor mínimo de $E(m)$ e o próprio valor mínimo mais 10 dB, sendo Q dado pelo valor central da faixa de pico máximo do histograma. Este procedimento é demonstrado na Seção 2.4.1.

Na primeira etapa do recorte são realizados os cálculos de energia para cada segmento, e depois realiza-se a equalização da energia com o ruído de fundo através da forma

$$\hat{E}(m) = \log[E(m)] - Q, \qquad m = 1, 2, \ldots, L \tag{2.3}$$

onde $E(m)$ é a energia em cada segmento m, L é o número total de segmentos e $\hat{E}(m)$ é a energia equalizada em cada segmento m. A utilização do logaritmo em (2.3) é para que não seja necessário modificar os valores dos limiares de energia sempre que forem calculados novos valores para Q, uma vez que a percepção auditiva se dá em escala logarítmica [3]. A vantagem do segundo método de obtenção de Q em relação ao primeiro, de acordo com [8], se deve ao fato deste método ser capaz de atualizar o valor de Q durante o processamento do recorte, sendo assim sensível a possíveis variações de energia do ruído. Outra vantagem é devido ao fato deste método não necessitar de um determinado número de amostras do ruído de fundo para obter Q. A desvantagem do segundo método se apresenta computacionalmente, pois o primeiro método realiza o cálculo de Q uma única vez e o segundo pode realizar tal cálculo várias vezes, dependendo do tamanho do sinal, e além disso, seu processamento executa maior número de operações que o primeiro método.

Durante a inicialização, carregam-se os seguintes limiares de energia: k_1 é

o limiar para obtenção da primeira estimativa do início de um pulso, k_2 é o limiar para obtenção tanto da segunda estimativa de início de um pulso quanto da primeira estimativa do final de um pulso, k_3 é o limiar para obtenção da segunda estimativa do final de um pulso e k_4 é o limiar que estipula a validade de um pulso. Os limiares de tempo (duração) são: T_1 que é usado para analisar qual estimativa a início de pulso será efetivamente utilizada, T_2 que é usado para analisar qual estimativa a final de pulso será considerada, T_3 que é usado para analisar se o pulso de energia encontrado é válido. Há ainda dois limiares, N_F e N_{sep}, usados para ordenar todos os pulsos previamente detectados com os limiares anteriores. O limiar N_F é usado para analisar se uma palavra recortada tem o tempo de duração válido e N_{sep} é usado para comparar o tempo de duração entre pulsos de energia e ordená-los.

A segunda etapa é a detecção de pulsos de energia. Para isto, inicia-se a análise de $\hat{E}(m)$ a partir do primeiro segmento do sinal até L. Durante esta "caminhada" analisa-se a existência de pulsos de energia, observando primeiramente a existência de um segmento que exceda o limiar k_1, como visto na Figura 2.5. Quando isto ocorrer marca-se A_1 como candidato a início de pulso detectado. Se a energia baixar de k_1 sem antes exceder de k_4, descarta-se A_1 e a análise recomeça, caso contrário, procura-se um segmento que exceda k_2 e marca-se A_2 como outro possível candidato a início de pulso. Se o tempo de duração entre A_1 e A_2 exceder T_1, então marca-se A_1 como início de pulso, pois considera-se a existência de uma fricativa neste intervalo de tempo, caso contrário, marca-se A_2 como início de pulso, pois considera-se a existência de alguma interferência, ou seja, um "click".

Continuando a analisar a energia normalizada do sinal, observa-se se a energia ultrapassa k_4. Caso isto ocorra, pode-se considerar que realmente se trata de um pulso de energia, e não uma perturbação sonora devido a algum ruído ambiente, onde o pulso seria descartado.

Dando continuidade ao processamento, analisa-se a queda da energia que já havia ultrapassado k_4 para k_2, marcando A_3 como candidato a fim de pulso. Continua-se a análise da queda da energia notando se ocorre uma queda passando por k_3 e marca-se A_4 como outro candidato a fim de pulso. Se o tempo de duração

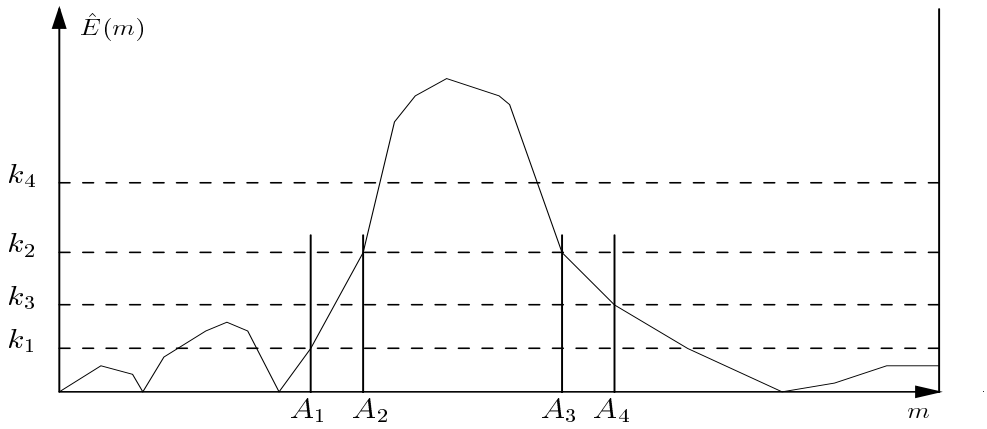

Figura 2.5: Ilustração da etapa de detecção de pulsos.

entre A_3 e A_4 for maior que T_2, então marca-se A_3 como final de pulso, pois considera-se a presença de sopro no final do pulso. Caso o intervalo seja menor que T_2, marca-se A_4 como final de pulso, pois considera-se a presença de fricativa neste intervalo.

A partir deste momento têm-se os pontos $P_B(i)$ e $P_E(i)$ que são respectivamente os pontos inicial e final do pulso de índice i. Caso o tempo de duração entre $P_B(i)$ e $P_E(i)$ seja menor que T_3, descarta-se o pulso obtido, pois considera-se a ocorrência de uma interferência, como por exemplo, uma batida de porta. Estes procedimentos acima devem ser executados até que se percorra o sinal por completo (até o segmento de número L), havendo no final um certo número de pulsos de energia encontrados no sinal.

A terceira etapa é a ordenação dos pulsos de tal forma que a correta "união" dos pulsos formará o recorte "ótimo" da palavra. Parte-se do pulso de maior energia e calcula-se a distância entre o ponto inicial deste pulso e o ponto final do pulso anterior. Caso essa distância seja menor que N_{sep}, agrega-se o pulso anterior ao pulso de maior energia e refaz-se a análise para o pulso anterior a este, obtendo assim o ponto mais à esquerda do sinal como início da palavra.

O mesmo se faz para obter o ponto mais a direita do sinal, marcando-o deste modo como final da palavra. Se a distância entre os pontos inicial e final for menor que N_F, então descarta-se a palavra.

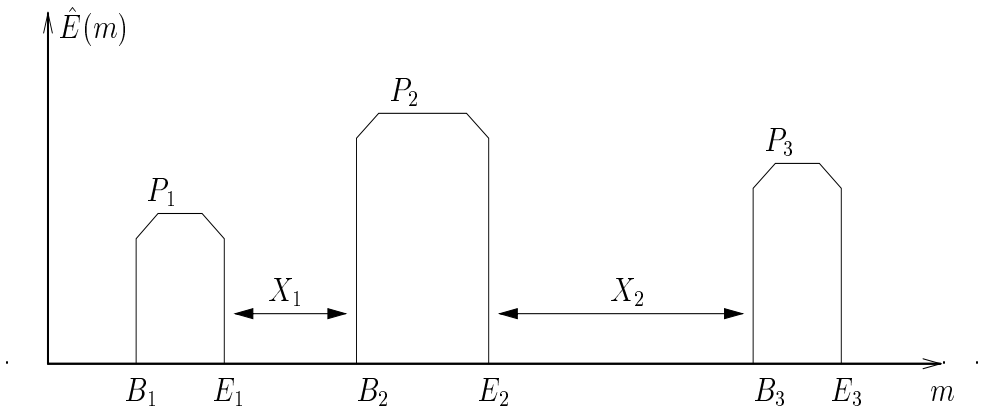

Figura 2.6: Exemplo da etapa de ordenação de pulsos.

Caso haja necessidade de se obter um outro recorte menos provável da palavra, analisa-se a distância do início da palavra recortada e fim do próximo pulso a esquerda e compara-se com a distância entre o final da palavra recortada com o início de um próximo pulso a direita, a menor das distâncias terá seu respectivo pulso incluído no recorte desta nova palavra. Este processo de inclusão de pulsos ocorre recursivamente enquanto houver pulsos a serem analisados e a necessidade de se obter novo recorte ("menos provável"). Quando terminarem os pulsos faz-se o procedimento inverso, excluindo os pulsos de forma análoga à inclusão, ou seja, eliminando o pulso que apresenta maior distância, também recursivamente até que seja atendida algum condição de término.

Para melhor compreender tal procedimento, será dado o exemplo da Figura 2.6. Observando a figura e assumindo que P_2 seja o pulso de maior energia e o intervalo $[B_2, E_2]$ seja maior que N_F. Então se $X_1 \leq N_{sep} < X_2$, o recorte mais provável será dado por $[B_1, E_2]$. Caso $X_1 \leq N_{sep}$ e $X_2 \leq N_{sep}$, o recorte mais provável será $[B_1, E_3]$. Caso $N_{sep} < X_1$ e $N_{sep} < X_2$ o recorte mais provável será $[B_2, E_2]$. Por fim, se $X_2 \leq N_{sep} < X_1$, o recorte mais provável será $[B_2, E_3]$.

2.4.1 Pseudo-código

Inicialmente carregam-se os valores dos limiares obtidos heuristicamente onde, $k_1 = 0.01$, $k_2 = 0.015$, $k_3 = 0.012$, $k_4 = 0.02$, $T_1 = 3$, $T_2 = 3$, $T_3 = 5$, $N_{sep} = 15$ e $N_F = 10$ [12]. Depois calcula-se o valor Q, que pode ser obtido aplicando-se o logaritmo em μ_e calculado de forma idêntica ao algoritmo 1, ou seja, $Q = \log_{10}(\mu_e)$.

A forma adaptativa de se calcular Q consiste em segmentar o sinal de voz original $s(n)$ com $n = 1, 2, \ldots, N_s$ através de L janelas retangulares sem superposição de tamanho N, de tal modo que o sinal segmentado será dado por $ss(m, n)$ onde, $m = 1, 2, \ldots, L$, $n = 1, 2, \ldots, N$. Depois calcula-se o logaritmo da energia em cada segmento $\log_{10}(E(m))$ onde, $m = 1, 2, \ldots, L$ e calcula-se o valor mínimo de energia do sinal $vmin = \min_{m=1,\ldots,L}\{\log_{10}(E(m))\}$. Faz-se um histograma com dez faixas do logaritmo da energia do sinal dentro do intervalo $[vmin, vmin + 10$ dB$]$, onde Q será o valor central da faixa que tiver maior concentração de energia.

Na Tabela 2.2, $P_B(i)$ e $P_E(i)$ representam os segmentos inicial e final do $i-$ésimo pulso de energia, onde $i = 1, 2, \ldots, npulse$, sendo $npulse$ o número total de pulsos de energia. Já $I(x)$ e $F(x)$ representam os segmentos inicial e final do $x-$ésimo recorte realizado, onde $x = 1, 2, \ldots, nrec$, sendo $nrec$ o número total de recortes realizados (ordem decrescente de probabilidade).

Tabela 2.2: Pseudo-código do algoritmo 2.

Inicialização:	
	Carregar os limiares: $k_1, k_2, \ldots, k_4, T_1, T_2, T_3, N_{sep}$ e N_F.
	Segmentar $s(n)$.
	Obter Q.
1^a Etapa:	
	Calcular $\hat{E}(m)$, para $m = 1, 2, \ldots, L$, pela equação (2.3).
2^a Etapa:	
	Obter $P_B(i)$ e $P_E(i)$, para $i = 1, 2, \ldots, npulse$.
3^a Etapa:	
	Obter $I(x)$ e $F(x)$, para $x = 1, 2, \ldots, nrec$.

2.5 Algoritmo 3

O algoritmo 3 [5] utiliza o método k-médias modificado [9] para realizar na detecção dos extremos da palavra. A partir da representação paramétrica dos segmentos do ruído ambiente obtidos no momento da gravação, calcula-se k grupos de ruído e seus vetores mais representativos através do algoritmo k-médias modificado [9]. Com estes vetores representativos e seus respectivos grupos faz-se a procura no sinal de segmentos que não façam parte de nenhum dos grupos encontrados, podendo tais segmentos satisfazerem as condições para pontos de recorte.

Este algoritmo se mostra menos sensível ao ruído por não ter limiares especificamente ajustados para a detecção da presença de fala no sinal analisado. A escolha da representação paramétrica dos segmentos e o número de parâmetros são mais dependentes da técnica de reconhecimento utilizada do que propriamente da técnica de recorte.

O algoritmo 3 pode ser dividido em três etapas bem distintas, praticamente não havendo a etapa de inicialização.

Na primeira etapa segmenta-se o sinal já utilizando as janelas e os parâmetros necessários ao reconhecimento. De acordo com [5], 90 segmentos que representem a gravação do ruído ambiente já são suficientes para um bom cálculo dos k grupos de ruído. Estes segmentos de ruído, em geral, estão representados nas primeiras amostras do sinal original que sejam equivalentes a pelo menos 90 segmentos do sinal.

Na segunda etapa, observando a representação paramétrica dos segmentos de ruído como vetores de parâmetros, aplica-se tais vetores ao algoritmo k-médias modificado [9] com o intuito de agrupar o universo dos vetores de ruído em k grupos. Este algoritmo irá gerar os k grupos, por intermédio dos seus k vetores centrais ($centro(i)$, onde $i = 1, 2, \ldots, k$) e pelo raio R de cada grupo, como é mostrado na Figura 2.7. Menciona-se em [5] que $k = 3$ já é suficiente para a realização de um recorte eficiente. Os raios R de cada grupo são dados por

$$R_n = 1.5Dmax - \frac{DIn}{2} \tag{2.4}$$

onde, R_n é o raio do grupo n, $Dmax$ é o diâmetro do grupo de maior dispersão, sendo o diâmetro dado pela maior distância entre dois elementos do grupo, e DIn é a distância intra-grupo do grupo n, que é dada pela média das distâncias entre o *centro* e os elementos pertencentes ao grupo [9].

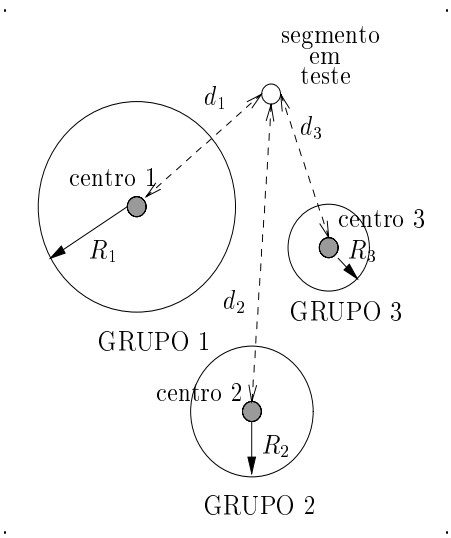

Figura 2.7: Representação dos grupos de ruído.

Na terceira etapa é realizada a procura dos pontos de recorte, ou seja, a partir do primeiro segmento caminha-se em direção ao extremo final do sinal calculando as distâncias d_n de cada segmento do sinal para todos os n vetores de parâmetros que representam os centros dos k grupos, como é mostrado na Figura 2.7. Se $d_n < R_n$, então, o segmento em teste é considerado como "segmento de ruído", caso contrário, o segmento em teste é considerado como "segmento de voz".

Para se determinar o extremo inicial de recorte, procura-se um "segmento de voz" a partir do primeiro segmento do sinal. Quando este for encontrado, testa-se os próximos 9 segmentos. Se entre estes houver 4 ou mais "segmentos de voz", então marca-se o primeiro segmento encontrado como ponto inicial de recorte. Caso não haja 4 ou mais "segmentos de voz", então continua-se o procedimento recursivamente até que o ponto inicial de recorte seja encontrado, ou o sinal seja descartado (quando se alcança o extremo final sem encontrar o ponto inicial de recorte).

Para se determinar o extremo final de recorte, continua-se a caminhar no sen-

tido do segmento final do sinal, agora procurando um "segmento de ruído". Quando este for encontrado, testa-se os próximos 10 segmentos, caso 4 ou mais segmentos entre estes sejam "segmentos de ruído", então marca-se o último segmento destes 10 como ponto final de recorte. Caso não ocorra 4 ou mais "segmentos de ruído", então continua-se recursivamente este procedimento até que o ponto final seja encontrado, ou o sinal seja descartado por alcançar o último segmento do sinal sem obter o ponto final de recorte.

2.5.1 Pseudo-código

Inicialmente tem-se o sinal original $s(n)$ com $n = 1, 2, \ldots, N_s$. Precisa-se ter as especificações de tipo de janela usada na segmentação, o tamanho das janelas N, quais e quantos parâmetros p serão utilizados para representar um segmento de voz. No trabalho apresentado em [5] foi utilizada janela de Hamming com $N = 20$ ms e 50% de superposição na segmentação de $s(n)$. Os parâmetros extraídos de cada segmento foram os mel-cepestrais com $p = 12$, porém cada segmento foi representado ainda por 1 coeficiente de energia. A representação final de cada segmento t foi dada pela junção dos 13 coeficientes dos segmentos $(t-1)$, t e $(t+1)$, gerando um total de 39 coeficientes por segmento.

Na Tabela 2.3, I e F representam os segmentos inicial e final de recorte obtidos pela comparação dos segmentos com os k grupos de ruído.

Tabela 2.3: Pseudo-código do algoritmo 3.

1^a Etapa:	
	Segmentar $s(n)$.
	Extrair o vetor de parâmetros para cada segmento.
2^a Etapa:	
	Obter $centro$, $Dmax$ e DIn para os k grupos de ruído.
3^a Etapa:	
	Comparar os segmentos com os k grupos de ruído.
	Obter os segmentos de recorte I e F

2.6 Algoritmo 4

O algoritmo 4 é um algoritmo proposto neste trabalho, sendo baseado nos conceitos utilizados pelos algoritmos 2 e 3 [8, 5]. Do mesmo modo que o algoritmo 3, este algoritmo utiliza o método k-médias modificado [9] para obter k grupos de ruído e seus vetores mais representativos. Com estes vetores representativos e seus respectivos grupos faz-se a procura no sinal de pulsos de informação de fala, analogamente à obtenção de pulsos de energia do algoritmo 2 [8]. Após a obtenção dos "pulsos de voz" analisa-se a separação entre os pulsos de modo a concatená-los ou não, de forma análoga à ordenação mais provável dos pulsos realizada no algoritmo 2.

Este algoritmo incorpora vantagens do algoritmo 3, como a pequena sensibilidade em relação ao ruído, e utiliza a metodologia do algoritmo 2 para o critério de decisão, ou seja, utiliza o critério aplicado no algoritmo 3 para observar a presença de fala no sinal analisado e utiliza o critério aplicado no algoritmo 2 para a obtenção dos pontos de recorte.

A implementação do algoritmo 4 pode ser dividida em quatro etapas, sendo as duas primeiras idênticas às descritas no algoritmo 3, exceto pelo fato de que devemos ainda definir os limiares T_3, N_{sep} e N_F, já descritos no algoritmo 2.

Na terceira etapa é realizada a detecção dos "pulsos de voz", ou seja, a partir do primeiro segmento caminha-se em direção ao outro extremo do sinal (extremo final) calculando as distâncias d de cada segmento do sinal para todos os vetores de parâmetros que representam os centros dos k grupos, como é mostrado na Figura 2.7. Se $d_n < R_n$, então, o segmento em teste é considerado como "segmento de ruído", caso contrário, o segmento em teste é considerado como "segmento de voz".

Ao encontrar um primeiro "segmento de voz", marca-se este como possível candidato a segmento inicial de pulso. Continua-se a procura até encontrar um "segmento de ruído". Ao ser encontrado, faz-se um teste de continuidade, analisando os próximos 5 segmentos a partir do "segmento de ruído". Se dentre os 5 segmentos testados um número maior ou igual a 3 for de "segmentos de voz", a busca pelo segmento final do "pulso de voz" continua a partir do último segmento dentre os 5 testados. Caso contrário, marca-se como segmento final do pulso, o último "seg-

mento de ruído" antes da realização do teste de continuidade. Após marcados os segmentos inicial e final do pulso, testa-se a validade do pulso. Se o comprimento do pulso for maior que T_3, então o pulso é válido, caso contrário, descarta-se o pulso encontrado. Este procedimento é realizado recursivamente até se alcançar o último segmento do sinal de fala, gerando no final $P_B(i)$ e $P_E(i)$ que representam os segmentos inicial e final do $i-$ésimo "pulso de voz" encontrado.

A quarta etapa é análoga à ordenação dos pulsos no algoritmo 2, de tal forma que a correta "união" dos pulsos formará o recorte "ótimo" da palavra. Parte-se do pulso de maior comprimento $(P_E(i) - P_B(i))$ e calcula-se a distância entre o ponto inicial deste pulso e o ponto final do pulso anterior. Caso essa distância seja menor que N_{sep}, agrega-se o pulso anterior ao pulso de maior comprimento e refaz-se a análise para o pulso anterior a este, obtendo assim o ponto mais à esquerda do sinal, I, como início da palavra. Procedimento equivalente é realizado com o intuito de se agregar pulsos a direita do pulso de maior comprimento, e por conseguinte obter o ponto mais a direita do sinal, F, como fim da palavra. Caso o comprimento da palavra $(F - I)$ seja maior que N_F, então o recorte é válido, caso contrário, invalida-se o recorte gerando uma mensagem de erro.

2.6.1 Pseudo-código

Inicialmente tem-se o sinal original $s(n)$ com $n = 1, 2, \ldots, N_s$. Precisa-se ter as especificações de tipo de janela usada na segmentação, o tamanho das janelas N, quais e quantos parâmetros p serão utilizados para representar um segmento de voz. Tipicamente utiliza-se janela de Hamming com $N = 20$ ms e 50% de superposição na segmentação de $s(n)$. Os parâmetros extraídos de cada segmento foram os mel-cepestrais com $p = 12$, porém cada segmento foi representado por 13 coeficientes relativos aos 12 mel-cepestrais mais 1 coeficiente de energia. A representação final de cada segmento t foi dada pela junção dos 13 coeficientes dos segmentos $(t - 1)$, t e $(t + 1)$, gerando um total de 39 coeficientes por segmento. Os valores utilizados nos limiares T_3, N_{sep} e N_F foram respectivamente 5, 20 e 15 segmentos.

Na Tabela 2.4, I e F representam os segmentos inicial e final de recorte

obtidos pela comparação dos segmentos com os k grupos de ruído, $P_B(i)$ e $P_E(i)$ são os segmentos inicial e final do $i-$ésimo pulso encontrado e $npulse$ o número total de pulsos encontrados.

Tabela 2.4: Pseudo-código do algoritmo 4.

1^a Etapa:	
	Segmentar $s(n)$.
	Extrair o vetor de parâmetros para cada segmento.
	Carregar os limiares T_3, N_{sep} e N_F.
2^a Etapa:	
	Obter $centro$, $Dmax$ e DIn para os k grupos de ruído.
3^a Etapa:	
	Obter $P_B(i)$ e $P_E(i)$, sendo $i = 1, \ldots, npulse$.
4^a Etapa:	
	Obter o pulso de maior comprimento.
	Ordenar os pulsos através de N_{sep}, e obter I e F.

2.7 Outros algoritmos de recorte

Existem outros algoritmos projetados com o intuito de se melhorar o recorte das palavras, como por exemplo os dados em [13] e [14] que utilizam respectivamente wavelets e um conceito diferente de energia. Estes dois algoritmos utilizam energia do sinal "modificada" através de métodos específicos para se obter um recorte mais eficaz da palavra pronunciada. O trabalho apresentado em [13] utiliza wavelets para analisar o sinal de voz, e se mostrou bastante eficiente quando comparado com um método baseado no algoritmo 1 e com o método proposto em [14].

Outro trabalho que utiliza uma forma diferente de abordar o recorte de palavras é apresentado em [15], onde se menciona o artigo [16] que introduz uma técnica de *clustering* ao algoritmo de DTW, de modo a permitir sua aplicação em algoritmos tanto de recorte como de *word spotting*. Também podemos citar o recente

trabalho [17].

2.8 Resultados

Nesta seção são apresentados alguns resultados de diferentes algoritmos de recorte comparando seus segmentos iniciais e finais, sendo também realizadas análises quanto à sensibilidade dos algoritmos perante as perturbações sonoras mencionadas na Seção 2.2.

2.8.1 Teste relativo às perturbações sonoras

Nestes testes são realizadas análises da eficiência do recorte em relação às perturbações descritas na seção 2.2. Os testes são realizados com três sinais diferentes, onde foram inseridos "click", antes e depois da palavra pronunciada (dígito "um"), e sopro, após a palavra pronunciada (dígito "oito").

O algoritmo 1 apresentou bastante sensibilidade a estes tipos de perturbações, pois elas, em geral, possuem níveis de energia comparáveis ao nível de energia da palavra pronunciada, ocasionando erro já na obtenção de segmentos de recorte utilizando o limiar $le1$. De fato, o algoritmo 1 se mostrou ineficiente na realização do recorte, pois não foi capaz de separar o "click" e o sopro da palavra.

O algoritmo 2 apresentou bastante eficiência em eliminar o "click", mas não foi eficiente na eliminação do sopro, devido a este tipo de perturbação ter duração e energia equivalentes às da palavra.

O algoritmo 3 se mostrou parcialmente eficiente, sendo capaz de eliminar o "click", mas não sendo capaz de eliminar o sopro da palavra. O algoritmo 3 reconhece como "segmento de voz" os segmentos que compreendem o "click" e o sopro, sendo que, o critério para validar os pontos de recorte deste algoritmo, foi capaz de eliminar o "click" mas não foi capaz de eliminar o sopro do recorte. Este problema se deve à duração das perturbações e ao tempo de separação entre elas e a palavra.

Os resultados de recorte apresentados pelo algoritmo 4 perante as pertur-

bações foi idêntico aos resultados dos algoritmos 2 e 3. Ele foi capaz de eliminar o "click" e não foi capaz de eliminar o sopro. Como este algoritmo tem um critério de validação dos pontos de recorte semelhante ao critério de validação dos pulsos de energia utilizado no algoritmo 2, ele acabou gerando o mesmo erro que o algoritmo 2 na eliminação do sopro.

2.8.2 Teste relativo aos pontos de recorte

Nesta seção serão apresentados testes comparando os quatro algoritmos de recorte apresentados com o recorte manual, tendo sido os sinais gravados em 8 kHz de freqüência de amostragem, e utilizando-se segmentos de 20 ms com 50% de superposição. Os sinais utilizados nos testes foram os dígitos "um", "dois", "três", "quatro", "cinco", "seis", "sete", "oito", "nove" e "zero".

Em [4] foram realizados testes de reconhecimento referentes a recortes "manuais" de algumas palavras, os quais na média chegavam a 93% de acerto, depois foram alterados os pontos de recorte inicial e final, adicionando ou subtraindo no máximo dez segmentos destes pontos de recorte. O resultado do trabalho mostra que uma variação de dois segmentos em qualquer um dos pontos (inicial ou final), tendo como referência os pontos de recorte "manual", ocasiona uma redução de 2% no percentual de acerto; e variações de dez segmentos podem ocasionar reduções da ordem de 30% do percentual de acerto no reconhecimento.

A Tabela 2.5 compara os segmentos inicial e final da palavra recortada com o recorte manual da mesma. Nesta tabela a última linha mostar o valor médio de segmentos errados, ou seja, a variável *erro* é dada pelo somatório do módulo da diferença entre o valor do segmento encontrado com o recorte manual e o valor do segmento encontrado com outro algoritmo de recorte, dividindo o somatório pelo número de dígitos usados neste teste. Os segmentos inicial e final obtidos com o recorte manual são dados pela média de três medições realizadas em cada sinal de teste.

Considerando os resultados da Tabela 2.5, nota-se que o algoritmo 4 apresentou o recorte mais próximo do recorte manual, sendo então o que possivelmente

Tabela 2.5: Pontos de recorte usando as diferentes técnicas apresentadas.

	segmentos iniciais					segmentos finais				
	algoritmos					algoritmos				
dígito	manual	1	2	3	4	manual	1	2	3	4
"um"	82	84	83	82	82	112	109	113	111	112
"dois"	65	71	66	70	70	108	110	101	105	107
"tres"	59	60	50	61	59	92	97	91	90	92
"quatro"	33	33	33	33	33	60	69	70	52	60
"cinco"	40	16	41	43	43	70	69	75	60	70
"seis"	37	21	37	37	37	68	71	66	67	68
"sete"	32	15	33	32	32	61	79	60	54	60
"oito"	47	48	41	48	47	75	97	98	74	76
"nove"	52	54	13	53	53	87	86	119	76	86
"zero"	73	68	73	77	77	100	116	101	96	100
erro	0	7.4	5.8	1.6	0.8	0	8.0	8.3	4.8	0.4

irá gerar o melhor percentual de reconhecimento [5, 6]. Em geral, os algoritmos apresentados mostraram resultados bastante coerentes, porém os algoritmos 1 e 2, devido à maior sensibilidade relativa dos seus limiares, apresentaram resultados piores quando comparados aos algoritmos 3 e 4.

2.9 Conclusão

A importância de um "bom" algoritmo de recorte pode ser ressaltada pelos resultados relativos a [4], já mencionados na seção anterior. Comparando os valores apresentados em [4] com os valores mostrados na Tabela 2.5, onde são analisados diferentes algoritmos de recorte observando seus pontos de recorte inicial e final, reforça-se uma vez mais o quão crítico pode ser o recorte para um sistema de reconhecimento. O algoritmo 1 apresentou variações da ordem de 25 segmentos nos

pontos de recorte em relação ao recorte "manual", o que certamente resultaria em percentuais de reconhecimento muito a baixo do que o sistema seria capaz de gerar com o recorte manual.

Concluindo, este trabalho apresenta de forma objetiva a influência de diferentes algoritmos de recorte em relação a sistemas de reconhecimento de palavras isoladas. É ainda proposto um novo algoritmo que se mostrou mais eficiente que outros implementados neste trabalho, considerando as especificações utilizadas no teste.

Capítulo 3

Revisão de algoritmos de extração de parâmetros

3.1 Introdução

Na etapa de extração de parâmetros, ocorre uma diminuição da seqüência representativa do sinal. Por exemplo, um segmento de sinal de fala de 20 ms amostrado a 8 kHz é representado por uma sequência numérica de 160 pontos e sua representação paramétrica pode ser dada por uma seqüência de 15 valores. Esta observação se mostra clara, considerando que o interesse desejado para determinados sistemas não é comparar sinais integralmente, e sim algumas características implícitas neles.

Os parâmetros mais usados em reconhecimento são, em ordem histórica, os LPC (*linear predictive code*), cepestrais, mel-cepestrais e PLP (*perceptual linear predictive*). Estes coeficientes são utilizados para representar o sinal de fala, pois, além de diminuir o tamanho do vetor que representa o segmento do sinal de fala, eles mantêm a informação "mínima" necessária capaz de diferenciar os vários segmentos existentes.

Neste capítulo será disticutido o uso destes diferentes tipos de coeficientes em sistemas de reconhecimento, procurando ressaltar suas principais diferenças e respectivas vantagens. Resultados comparativos em aplicações de reconhecimento de voz utilizando os diferentes parâmetros descritos neste capítulo são apresentados

nos Capítulos 4, 5 e 6. Este capítulo descreve respectivamente os coeficientes LPC, cepestrais por definição, cepestrais extraídos dos coeficientes LPC, mel-cepestrais e PLP.

3.2 Coeficientes LPC

A codificação por predição linear (*linear predictive coding*, LPC) é baseada no modelo simplificado de produção da fala visto na Figura 3.1 [3]. Este modelo considera que a fala pode ser representada como um processo estocástico estacionário por partes. Com isto, assume-se que a fala pode ser gerada a partir da filtragem, no caso, por um filtro puramente recursivo, de um certo sinal de excitação.

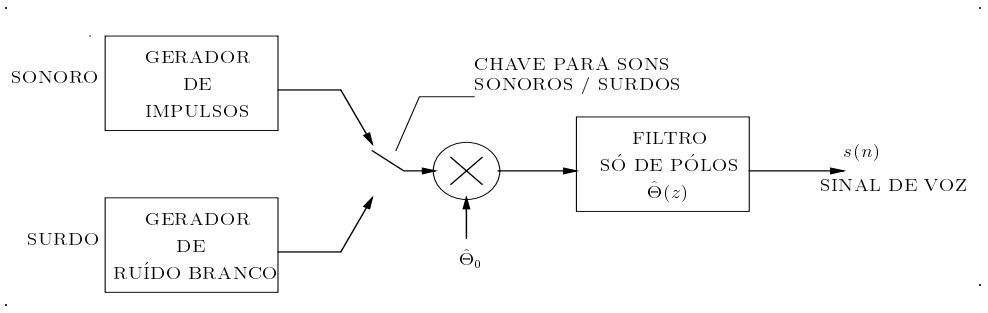

Figura 3.1: Modelo LPC de produção de voz.

O modelo LPC considera ainda que a fala é constituída de duas classes de sons: surdos e sonoros, visualizados respectivamente nas Figuras 3.2 e 3.3. Os sons surdos são caracterizados por uma baixa correlação entre as amostras, correspondendo assim a sons tipicamente ruidosos como aqueles associados a fricativas. Por outro lado, os sons sonoros são caracterizados por uma forte correlação entre suas amostras apresentando uma forma de onda quase-periódica, como facilmente constatado na Figura 3.3. No modelo LPC esta diferenciação entre sons se dá em termos da excitação de um filtro digital. Sons surdos são gerados por uma excitação do tipo ruído branco, e sons sonoros são gerados com uma excitação do tipo trem de pulsos.

A utilização de um filtro puramente recursivo se justifica por dois motivos. O primeiro é devido à necessidade matemática, pois este modelo pode ser facilmen-

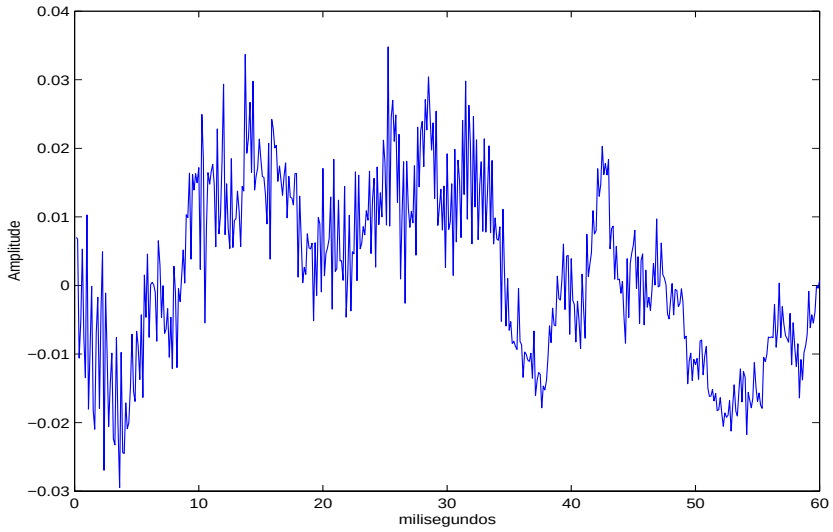

Figura 3.2: Exemplo de som surdo.

te resolvido através de um sistema de equações lineares, como visto adiante. O segundo é devido a boa modelagem dos pólos, possibilitando assim uma eficiente representação dos sinais de fala sonoros [3].

A partir do modelo simplificado da Figura 3.1, tem-se

$$S(z) = \hat{\Theta}_0 \hat{\Theta}(z) U(z) \tag{3.1}$$

onde $S(z)$ e $U(z)$ são respectivamente as transformadas $\mathcal{Z}$ de $s(n)$ que é o sinal de fala, $u(n)$ que é a excitação utilizada no modelo, e $\hat{\Theta}_0$ é a constante de ganho relacionada ao modelo. Além disso, $\hat{\Theta}(z)$ é o filtro digital usado para representar as características do trato vocal, sendo dado por

$$\hat{\Theta}(z) = \frac{1}{1 - \sum_{i=1}^{M} \hat{a}(i) z^{-1}} \tag{3.2}$$

onde M é a ordem do modelo LPC utilizado, $\hat{a}(i)$ são os coeficientes LPC estimados. A utilização de "$\hat{}$" sobre as variáveis mencionadas anteriormente indica que elas são estimativas de valores ideais.

No domínio do tempo temos

$$s(n) = \sum_{i=1}^{M} \hat{a}(i) s(n-i) + \hat{\Theta}_0 u(n) \tag{3.3}$$

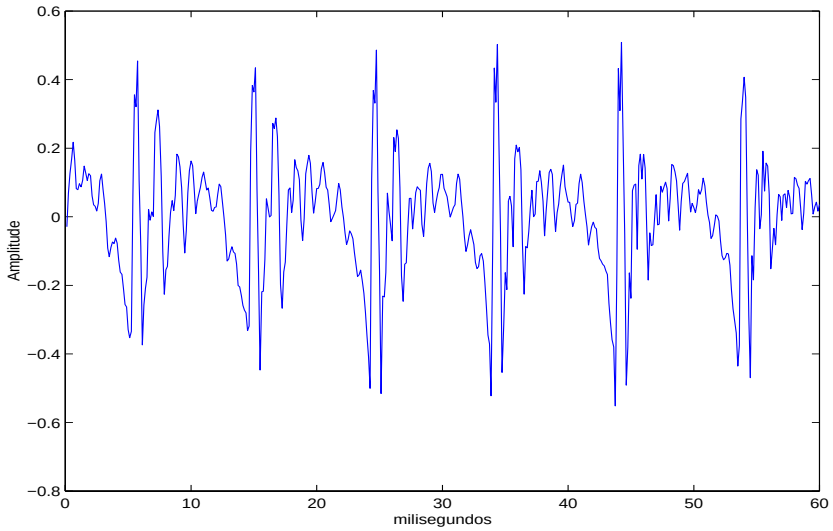

Figura 3.3: Exemplo de som sonoro.

Definindo o sinal de fala estimado pelo modelo, $\hat{s}(n)$, como sendo

$$\hat{s}(n) = \sum_{i=1}^{M} \hat{a}(i)s(n-i) \tag{3.4}$$

podemos formar o erro de predição dado por

$$e(n) = s(n) - \hat{s}(n) = s(n) - \sum_{i=1}^{M} \hat{a}(i)s(n-i) \tag{3.5}$$

Elevando ambos os lados da equação (3.5) ao quadrado, e calculando o valor esperado, tem-se o funcional

$$\mathcal{E}[e^2(n)] = \mathcal{E}[s^2(n) - 2s(n)\hat{s}(n) + \hat{s}^2(n)] \tag{3.6}$$

Para minimizar esta função objetivo, podemos derivá-la em relação aos coeficientes do modelo LPC, e igualar o resultado a zero, obtendo o seguinte sistema de equações lineares

$$\begin{bmatrix} r(0) & r(1) & \cdots & r(M-1) \\ r(1) & r(0) & \cdots & r(M-2) \\ \vdots & \vdots & \vdots & \vdots \\ r(M-1) & r(M-2) & \cdots & r(0) \end{bmatrix} \begin{bmatrix} \hat{a}(1) \\ \hat{a}(2) \\ \vdots \\ \hat{a}(M) \end{bmatrix} = \begin{bmatrix} r(1) \\ r(2) \\ \vdots \\ r(M) \end{bmatrix} \tag{3.7}$$

onde $r(k)$ para $k = 0, 1, \ldots, M$, é dado por

$$r(k) = \mathcal{E}[s(n)s(n-k)] \tag{3.8}$$

isto é, $r(k)$ é o valor da autocorrelação do processo $s(n)$ com *lag* k. A solução deste sistema é o conjunto de coeficientes de predição linear.

3.2.1 Algoritmo (Autocorrelação)

O sinal $s(n)$ é segmentado para que tal procedimento possa ser executado em tempo real e para que seus segmentos possam ser assumidos como sinais estacionários, condição implícita do modelo LPC. O algoritmo a seguir, será descrito para um único segmento de voz, de tamanho N.

i) Dado o segmento $x = [x(1)x(2)\ldots x(N)]$, calcular os valores de

$r(k) = \frac{1}{N}\sum_{n=1}^{N} x(n)x(n-k)$, para $k = 0, 1, \ldots M$.

ii) Montar o sistema da equação (3.7).

iii) Resolver o sistema pelo método Levinson-Durbin [3], determinando os coeficientes $\hat{a}(i)$, para $i = 1, 2, \ldots M$.

3.3 Coeficientes Cepestrais

A idéia da análise cepestral se baseia em representar o sinal de fala em um domínio onde seja possível uma melhor separação das componentes referentes ao trato vocal e à excitação. Observando a representação de um sinal de fala, temos que $s(n) = u(n) * \hat{\theta}(n)$ onde $s(n)$ é o sinal de fala discreto no tempo, $u(n)$ é a excitação, $\hat{\theta}(n)$ é a resposta ao impulso do trato vocal, e $*$ é o operador convolução. No domínio da freqüência teremos uma multiplicação dos espectros de $u(n)$ e de $\hat{\theta}(n)$, como visto na equação (3.1), não conseguindo facilmente separar as duas componentes. O ganho do modelo não foi considerado, pois ele é simplesmente um fator multiplicativo que não influência na separação das componentes de um sinal de fala.

A partir desta dificuldade surgiu o domínio cepestral real, definido por

$$c_s(k) = \mathcal{F}^{-1}\{\log|\mathcal{F}\{\hat{s}(n)\}|\} \tag{3.9}$$

de modo que

$$c_s(k) = c_u(k) + c_{\hat{\theta}}(k) \tag{3.10}$$

isto é, a representação de $s(n)$ no domínio cepestral, $c_s(n)$, é igual à soma das componentes associadas ao sinal de excitação, $c_u(n)$, e ao trato vocal, $c_{\hat{\theta}}(n)$, também no domínio cepestral.

Os coeficientes cepestrais relativos ao trato vocal situam-se na parte mais baixa do domínio cepestral, enquanto que os coeficientes relativos à excitação situam-se numa parte mais alta deste mesmo domínio, exceto pelo coeficiente zero que também faz parte da excitação [3]. Como pode ser observado na Figura 3.4, os coeficientes relativos à excitação impulsiva de sons sonoros aparecem no domínio cepestral como impulsos quasi-periódicos de amplitude decrescente, e os coeficientes relativos ao trato vocal se concentram nas amostras iniciais deste domínio.

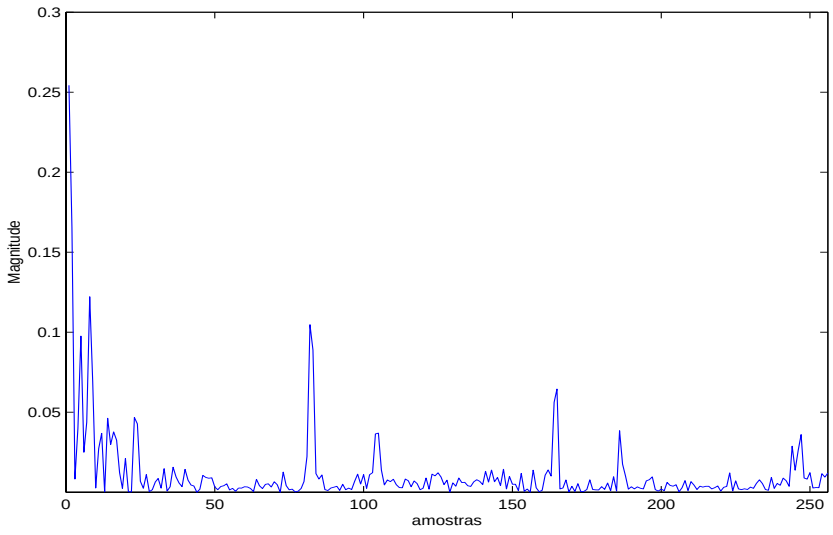

Figura 3.4: Separação das componentes do sinal de excitação e do trato vocal, no domínio cepestral (magnitude do sinal x número de amostras).

Para se separar os coeficientes do trato vocal no domínio cepestral, basta se fazer uma *liftragem* (filtragem no domínio cepestral) nas baixas baixas *qüefrências*

(freqüência no domínio cepestral). O tamanho do *liftro* necessário para capturar toda a informação referente ao trato vocal, $c_{\hat{\theta}}(n)$, é relacionado com a freqüência de *pitch* do sinal de fala no domínio cepestral. Na prática, costuma-se usar de 8 a 15 coeficientes cepestrais para representar cada segmento do sinal de voz.

O tipo de *liftro* mais usado [6, 3], é mostrado na Figura 3.5, onde se dá maior importância aos seus coeficientes centrais, sendo descrito por

$$l(n) = \begin{cases} 1 + \frac{L}{2}\sin\left(\frac{\pi n}{L}\right), & \text{n} = 0, 1, 2, \ldots, \text{L} \\ 0, & \text{qualquer outro valor de n} \end{cases} \tag{3.11}$$

onde L é o tamanho do *liftro*, sendo este o próprio número de coeficientes utilizad.

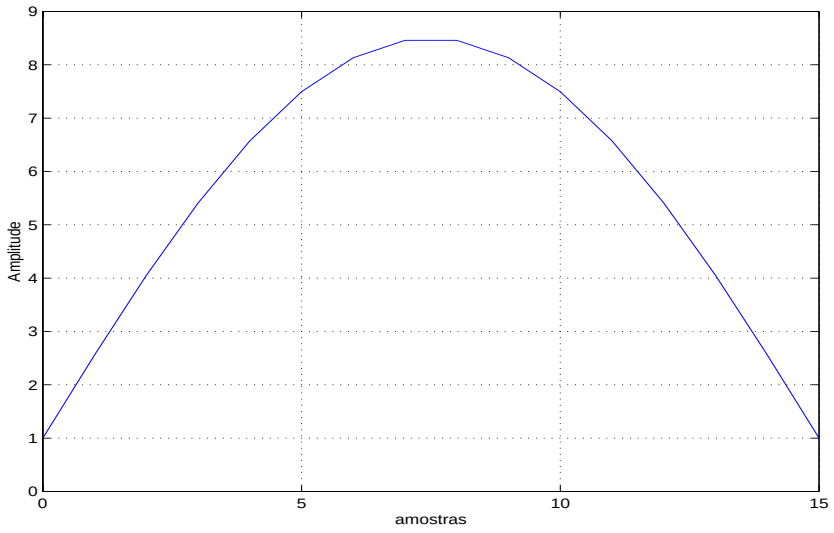

Figura 3.5: *Liftro* tipo senoidal de tamanho L =15.

3.3.1 Algoritmo

O sinal $s(n)$ será segmentado por janelas de tamanho N, e será utilizado um *liftro* de tamanho M, onde M é o próprio número de coeficientes cepestrais desejados por segmento. O algoritmo mostrado a seguir, descreve o cálculo para obtenção dos M coeficientes cepestrais para um segmento de $s(n)$.

i) Dado o segmento $x = [x(1) \ldots x(N)]$, calcular o valor absoluto da transformada de Fourier de x, dada por $|FFT\{x\}|$.

ii) Calcular o logaritmo e realizar a transformada inversa dos valores obtidos acima, sendo dado por, $c_x(n) = IFFT\{\log|FFT\{x\}|\}$.

iii) Realizar a *liftragem* em $c_x(n)$ usando um *liftro* de ordem M, obtido pela equação (3.11).

3.4 Coeficientes LPC-Cepestrais

Uma forma alternativa à dada em (3.9) para se calcular os coeficientes cepestrais, $c_{\hat{\theta}}(k)$, é a partir dos coeficientes LPC, $\hat{a}(k)$. Este método é apresentado em [3], sendo definido por

$$c_{\hat{\theta}}(k) = \begin{cases} \log\hat{\Theta}_0, & k = 0 \\ -(\frac{1}{2})\hat{a}(k) + (\frac{1}{4})\sum_{t=1}^{M}\left(\frac{t}{k}\right)c_{\hat{\theta}}(t)\hat{a}(k-t), & k = 1,\ldots,M \end{cases} \tag{3.12}$$

onde M é o número de coeficientes desejados, k representa o índice dos coeficientes e sendo o número de coeficientes LPC maior ou igual a M.

Uma vantagem deste método comparado com o método anterior de obtenção de coeficientes cepestrais é devido à complexidade computacional. Os coeficientes LPC-cepestrais são obtidos por intermédio de operações bem simples como mostrado na equação (3.12), enquanto que os coeficientes cepestrais como definidos na Seção 3.3 utilizam o cálculo da transformada de Fourier e sua inversa.

A desvantagem é devido à representação da fala pelo modelo de predição linear, ou seja, por um filtro puramente recursivo. Os sons surdos não são muito bem representados por este modelo [3], pois ao se observar o espectro deste tipo de som, se nota a presença de pontos nulos, o que intuitivamente daria a idéia de se incluir zeros no filtro que representa o trato vocal, o que não ocorre em $\hat{\Theta}(z)$.

3.4.1 Algoritmo

A partir dos coeficientes LPC, $\hat{a}(k)$, de um determinado segmento do sinal $s(n)$, e do ganho $\hat{\Theta}_0$ relativo ao filtro usado neste mesmo modelo, serão calculados os coeficientes cepestrais relativos a resposta ao impulso do trato vocal $c_{\hat{\theta}}(k)$. O número de coeficientes desejados por segmento será M.

i) Dado o segmento $x = [x(1) \dots x(N)]$, calcular $\hat{a}(1), \dots, \hat{a}(M)$ e $\hat{\Theta}_0$, por intermédio do algoritmo de extração dos coeficientes LPC, Seção 3.2.

ii) Calcular $c_{\hat{\theta}}(n)$ através da equação (3.12).

3.5 Coeficientes Mel-Cepestrais

A extração de coeficientes mel-cepestrais é baseada nos coeficientes cepestrais, exceto pela inclusão da percepção auditiva. Esta percepção é dada pela filtragem do sinal de fala, de modo a se relacionar a freqüência real com a percepção de tom realizada pelo ouvido humano, sendo isto chamado de filtragem perceptual.

A idéia da escala *mel* surgiu em 1940 [18], através de um experimento realizado por Volkman e Stevens. Neste experimento foi arbitrado que 1000 Hz seria igual a 1000 mels, e foi se alterando a freqüência de um determinado sinal e perguntando as pessoas que contribuiram para os testes quanto de aumento elas notavam na freqüência.

É interessante observar que a literatura relaciona mais de uma fórmula para aproximar esta relação. De acordo com [19] a relação é dada por

$$F_{mel} = \frac{1000 \ln\left(1 + \frac{F_{Hz}}{700}\right)}{\ln\left(1 + \frac{1000}{700}\right)} \tag{3.13}$$

Por outro lado, em [20] a relação é dada por

$$F_{mel} = 1000 \log_2(1 + F_{Hz}) \tag{3.14}$$

Como pode ser observado na Figura 3.6, as duas equações nos fornecem resultados bastante semelhantes, somente com grande discrepância em altas freqüências, o que não reflete um problema, uma vez que em sinais de fala a maior parte da energia se encontra em baixas freqüências. É importante também ressaltar a equação (3.15) apresentada em [3], que mostra uma relação errônea entre as freqüências (mel/Hz), a qual não atende nem à definição, onde é estabelecido que 1000 mel é igual a 1000 Hz.

$$F_{mel} = \frac{1000}{\log 2}\left[1 + \frac{F_{Hz}}{1000}\right] \tag{3.15}$$

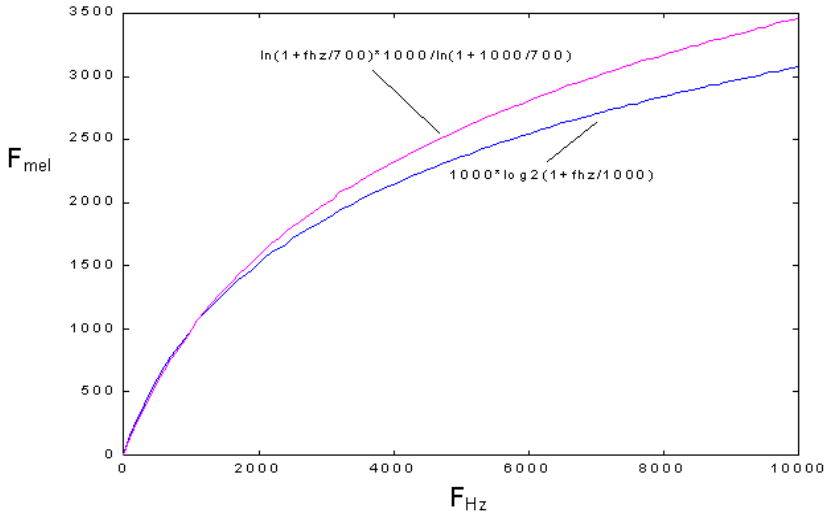

Figura 3.6: Relação entre F_{Hz} e F_{mel}.

Algumas variáveis básicas serão definidas para a apresentação das diferentes implementações do cálculo dos coeficientes mel-cepestrais realizado nos algoritmos de [3] e [6], respectivamente. A representação de um certo segmento do sinal de voz $s(n)$, é dada por $x(n)$, e sendo $X(k)$ a DFT deste segmento. São utilizados também filtros de banda crítica (filtros cb), vistos na Figura 3.7, cujo objetivo é realizar análises específicas em determinadas faixas de freqüência do sinal segmentado. Normalmente, são usados 20 filtros triangulares igualmente espaçados e com 50% de superposição em mels para a realização dos filtros cb [21, 3, 6]. As freqüências centrais destes filtros cb (em Hz) são dadas por

$$F_{c,i} = k_i \frac{F_s}{N'} \tag{3.16}$$

onde N' é o número de pontos usados no cálculo da DFT dos segmentos do sinal de fala, F_s é a freqüência de amostragem e k_i representa os pontos no eixo horizontal de $X(k)$, equivalentes à freqüência central do filtro i.

Existem diferentes formas de se obter os coeficientes mel-cepestrais, como o algoritmo de Deller et al. [3], o algoritmo de Rabiner et al. [6], o algoritmo que transforma de LPC para mel-cepestral recursivamente obtido em [1], e ainda, os coeficientes mel-cepestrais e os mel-cepestrais adaptativos encontrados em [22]. Também será apresentado um processamento auxiliar chamado de CMR (*cepestral mean removal*)

aplicado em geral a coeficientes mel-cepestrais. Este procedimento realmente aumenta consideravelmente o percentual de reconhecimento, como observado em [23].

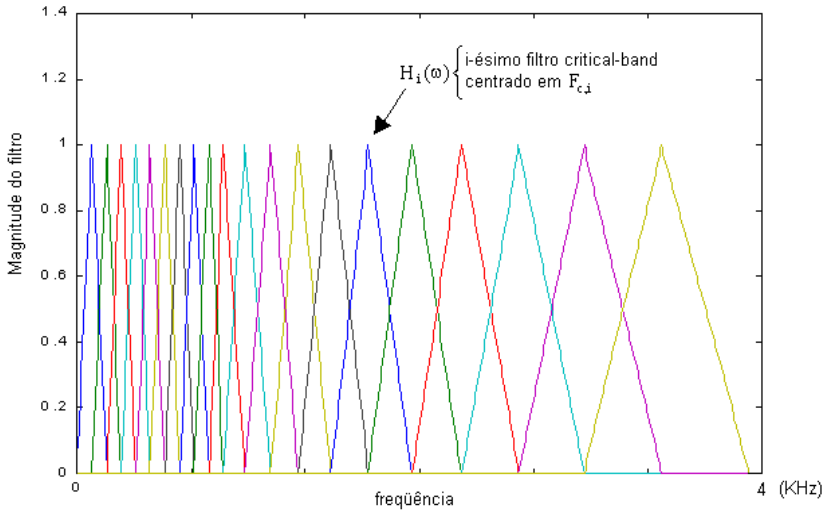

Figura 3.7: Filtros cb.

3.5.1 Algoritmo de Deller et al.

Nesta seção será apresentado o algoritmo para se calcular os coeficientes mel-cepestrais de acordo com [3].

Primeiramente, realiza-se a filtragem pelos filtros cb, dada por

$$Y(i) = \sum_{k=0}^{N'/2} \log |X(k)| H_i \left(k \frac{2\pi}{N'} \right) \tag{3.17}$$

sendo $Y(i)$ o resultado da filtragem de $X(k)$ pelos filtros cb, para $i = 1, \ldots, N_{cb}$, e N_{cb} o número de filtros cb utilizados.

Na segunda etapa, forma-se a seqüência auxiliar $\tilde{Y}(k)$, dada por

$$\tilde{Y}(k) = \begin{cases} Y(i), & k = k_i \\ 0, & \text{outros valores de } k \end{cases} \tag{3.18}$$

Na terceira e última etapa, realiza-se a transformada inversa dada pela equação

$$\tilde{c}_x(n) = \frac{2}{N'} \sum_{i=1,2,\ldots,N_{cb}} \tilde{Y}(k_i) \cos \left(k_i \frac{2\pi}{N'} n \right) \tag{3.19}$$

onde $\tilde{c}_x(n)$ são os coeficientes mel-cepestrais de $x(n)$, sendo n os índices dos coeficientes.

Os coeficientes $\tilde{c}_x(n,m)$ serão "filtrados" por um *liftro* passa-baixas, como já mencionado na Seção 3.3.

3.5.2 Algoritmo de Rabiner et al.

O cálculo dos coeficientes mel-cepestrais de [6] segue outra forma. Primeiramente é definido $\tilde{Y}_i$ como sendo o espectro de potência de um segmento do sinal de fala filtrado pelos filtros cb, sendo obtido pela equação

$$\tilde{Y}_i = \sum_{k=0}^{N'/2} |X(k)|^2 \, H_i \left(k \frac{2\pi}{N'} \right) \tag{3.20}$$

sendo $i = 1, \ldots, N_{cb}$, e N_{cb} o número de filtros cb utilizados. Na segunda e última etapa do cálculo,

$$\tilde{c}_x(n) = \sum_{i=1}^{N_{cb}} \log(\tilde{Y}_i) \cos \left[n \left(i - \frac{1}{2} \right) \frac{\pi}{N_{cb}} \right] \tag{3.21}$$

onde $n = 1, 2, \ldots, L$, sendo L o número desejado de coeficientes, e este precisa ser maior ou igual a M que será o número de coeficientes obtidos após a liftragem realizada pelo liftro em (3.11).

3.5.3 Algoritmo Recursivo

Em algoritmos LPC-MEL recursivos, os coeficientes mel-cepestrais são derivados a partir dos coeficientes LPC. O algoritmo aqui descrito foi proposto em [1], onde é demonstrado que se pode calcular os coeficientes mel-cepestrais a partir dos coeficientes LPC sem erro devido ao truncamento que ocorre em outros algoritmos LPC-MEL recursivos, aplicando-se primeiramente uma fórmula recursiva para transformação em freqüência, e posteriormente a fórmula recursiva que transforma coeficientes LPC em cepestrais, correspondendo à ordem inversa da utilizada em outros algoritmos.

Quando os coeficientes mel-cepestrais são derivados a partir dos LPC, pode-se primeiramente transformar os coeficientes LPC em coeficientes cepestrais, através da

fórmula recursiva de [24], e posteriormente realizar a transformação em freqüência, onde os coeficientes cepestrais são passados para o domínio mel-cepestral pela fórmula recursiva apresentada em [25]. O resultado da conversão LPC para cepestral é uma seqüência infinita que, na prática, para a obtenção dos coeficientes mel-cepestrais é truncada.

Na implementação recursiva inverte-se a seqüência de processamento, ou seja, primeiro é realizada a transformação em freqüência e depois é feita a transformação de coeficientes. Com isto é possível calcular os coeficientes mel-cepestrais sem o erro de truncamento mencionado.

Primeiramente será definida a representação mel-cepestral utilizada

$$\tilde{c}(m) = \frac{1}{2\pi j} \oint \log X(z) \tilde{z}^{m-1} \, d\tilde{z} \tag{3.22}$$

onde $X(z)$ é a transformada $\mathcal{Z}$ de $x(n)$ que representa um certo segmento do sinal de voz $s(n)$, e $\tilde{c}(m)$ representa os coeficientes mel-cepestrais extraídos do segmento $x(n)$. A seguir, será apresentada a representação inversa, ou seja, a representação de $X(z)$ a partir de $\tilde{c}(m)$, dada por

$$\log X(z) = \sum_{m=-\infty}^{\infty} \tilde{c}(m) \, \tilde{z}^{-m} \tag{3.23}$$

e por fim a transformação em freqüência, dada por

$$\tilde{z}^{-1} = \frac{z^{-1} - \alpha}{1 - \alpha z^{-1}}, \quad |\alpha| < 1 \tag{3.24}$$

onde se obtêm da referência [22] que esta transformação em freqüência só depende da freqüência de amostragem, e a transformação que relaciona uma freqüência de amostragem de 8 kHz com a escala mel é obtida com $\alpha = 0.31$.

Antes de dar continuidade ao cálculo dos coeficientes mel-cepestrais, serão definidos alguns parâmetros importantes: sejam M o número de coeficientes LPC, N o número de coeficientes mel-cepestrais, $\hat{a}(m)$ o coeficiente LPC de determinada ordem m, $\tilde{a}(m)$ o coeficiente $\hat{a}(m)$ transformado na freqüência, e $\hat{\Theta}_0$ o ganho do modelo de predição linear. A seguir será apresentada a fórmula que transforma coeficientes de predição linear em coeficientes mel-cepestrais recursivamente:

$$\tilde{a}^{(i)}(m) = \begin{cases} \hat{a}(-i) + \alpha\tilde{a}^{(i-1)}(0), & m = 0 \\ (1-\alpha^2)\tilde{a}^{(i-1)}(0) + \alpha\tilde{a}^{(i-1)}(1), & m = 1 \\ \tilde{a}^{(i-1)}(m-1) + \alpha(\tilde{a}^{(i-1)}(m) - \tilde{a}^{(i)}(m-1)), & m = 2, \ldots, N \end{cases} \tag{3.25}$$

onde $i = -M, \ldots, -1, 0$,

$$\tilde{K} = \frac{\hat{\Theta}_0}{\tilde{a}^{(0)}(0)}, \quad \tilde{a}(m) = \frac{\tilde{a}^{(0)}(m)}{\tilde{a}^{(0)}(0)}, \quad 1 \leq m \leq N \tag{3.26}$$

e por fim,

$$\tilde{c}(m) = \begin{cases} \log\tilde{K}, & m = 0 \\ -\tilde{a}(m) - \sum_{k=1}^{m-1} \frac{k}{m}\tilde{c}(k)\tilde{a}(m-k), & 1 \leq m \leq N \end{cases} \tag{3.27}$$

onde $\hat{a}(0) = 1$ e $\tilde{a}^{(m-1)}(m) = 0$ para $m = 1, 2, \ldots, M$.

As equações (3.25) e (3.26) utilizam como variáveis de entrada $\hat{a}(m)$, α, M, N e $\hat{\Theta}_0$ para calcular $\tilde{a}(m)$, que serão utilizados como entrada da equação (3.27) que gera os coeficientes mel-cepestrais $\tilde{c}(m)$.

A liftragem será aplicada aos N coeficientes $\tilde{c}(m)$, obtidos pela representação mel-cepestral de um segmento de $s(n)$.

3.5.4 *Cepestral mean removal*

O procedimento CMR (*cepestral mean removal*) é aplicado tipicamente a coeficientes mel-cepestrais, e consiste em se estimar a média de cada coeficiente mel-cepestral de uma determinada palavra, e depois subtrair de cada coeficiente esta média estimada. O objetivo desta técnica é tornar a representação do segmento de fala menos sensível a mudanças da média de longo termo do sinal, podendo este problema ser ocasionado por troca de microfones e/ou lentas variações no canal de comunicação [26]. Este procedimento foi capaz de ocasionar uma melhoria de cerca de 2% no percentual de reconhecimento de sistema que utiliza um classificador HMM, apresentado em [23].

3.6 Coeficientes PLP

Os coeficientes de predição linear perceptuais, PLP (*perceptual linear prediction*), apresentados em [2] utilizam algumas propriedades auditivas humanas, simuladas e aproximadas por filtros digitais, resultando num espectro do sinal de fala mais próximo do espectro percebido pelo nosso aparelho auditivo.

Os conceitos utilizados na obtenção destes coeficientes são: resolução espectral em banda crítica, diferença de sensibilidade do ouvido humano às diversas freqüências e relação não linear entre a intensidade do som real e a percebida. A aplicação destes conceitos ao espectro de potência do sinal resultará num espectro de voz parecido com o espectro percebido pelo ouvido humano, que será aproximado por um modelo puramente recursivo para obtenção dos coeficientes.

A obtenção dos coeficientes será apresentada a partir de um único segmento do sinal de fala $s(n)$ amostrado com freqüência Fs (em Hz).

Será definido $P(\omega)$ como o espectro de potência do segmento em questão, onde ω é a freqüência em rad/s. O espectro de potência $P(\omega)$ será transformado em freqüência de rad/s para Bark, cuja relação é descrita por

$$\Omega(\omega) = 6 \ln \left\{ \frac{\omega}{1200\pi} + \left[\left(\frac{\omega}{1200\pi} \right)^2 + 1 \right]^{0.5} \right\} \tag{3.28}$$

sendo Ω a representação da freqüência ω na escala Bark. Após a transformação em freqüência, $P(\Omega)$ será convoluído com $\Psi(\Omega)$ que é a "máscara" de filtros cb utilizados na implementação, a "máscara" esta apresentada e definida pela Figura 3.8 e equação (3.29). Todo este procedimento é equivalente à filtragem pelos filtros cb também realizada na obtenção dos coeficientes mel-cepestrais.

$$\Psi(\Omega) = \begin{cases} 0, & \text{para} \quad \Omega < -1.3 \\ 10^{2.5(\Omega+0.5)}, & \text{para} \quad -1.3 \leq \Omega \leq -0.5 \\ 1, & \text{para} \quad -0.5 < \Omega \leq 0.5 \\ 10^{-1.0(\Omega-0.5)}, & \text{para} \quad 0.5 < \Omega \leq 2.5 \\ 0, & \text{para} \quad \Omega > 2.5 \end{cases} \tag{3.29}$$

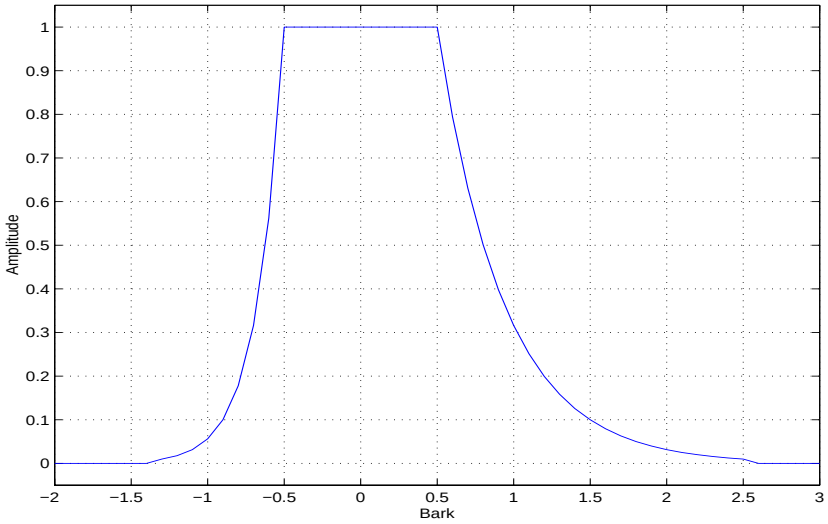

Figura 3.8: "Máscara" do filtro cb usado na obtenção dos coeficientes PLP.

A convolução de $P(\Omega)$ com $\Psi(\Omega)$ é dada por

$$\Theta(\Omega_i) = \sum_{\Omega=-1.3}^{2.5} P(\Omega - \Omega_i)\Psi(\Omega) \tag{3.30}$$

onde Ω_i são as freqüências centrais dos filtros cb (análogo aos mel-cepestrais). Neste método, estas freqüências são espaçadas em intervalos de aproximadamente 1 Bark. Por exemplo, um sinal amostrado em 10 kHz terá uma faixa de freqüência dinâmica de 0-5 kHz (0-16.9 Bark), da qual pode-se obter 18 amostras espaçadas de 0.994 Bark.

A próxima etapa é a realização da pré-ênfase em $\Theta[\Omega(\omega)]$ pela equação

$$\Xi[\Omega(\omega)] = EL(\omega)\Theta[\Omega(\omega)] \tag{3.31}$$

onde $EL(\omega)$ é uma aproximação da sensibilidade auditiva em diferentes freqüências e simula a sensibilidade auditiva num nível de aproximadamente 40 dB [2]. De acordo com [2], $EL(\omega)$ é dado por

$$EL(\omega) = \frac{[(\omega^2 + 56.8 \times 10^6)\omega^4]}{[(\omega^2 + 6.3 \times 10^6)^2(\omega^2 + 0.38 \times 10^9)]} \tag{3.32}$$

quando a faixa de freqüência de interesse chega até 5 kHz, já para freqüências acima de 5 kHz, esta curva é dada por

$$EL(\omega) = \frac{[(\omega^2 + 56.8 \times 10^6)\omega^4]}{[(\omega^2 + 6.3 \times 10^6)^2(\omega^2 + 0.38 \times 10^9)(\omega^6 + 9.58 \times 10^{26})]} \tag{3.33}$$

desta forma, sendo fundamental para sistemas que utilizem freqüências de amostragem maior que 10 kHz. A referência [27], relaciona um considerável número de sistemas que utilizam freqüências de amostragem maiores que 10 kHz.

Na Figura 3.9 é apresentado o gráfico de $EL(\omega)$, numa faixa de 0-4 kHz.

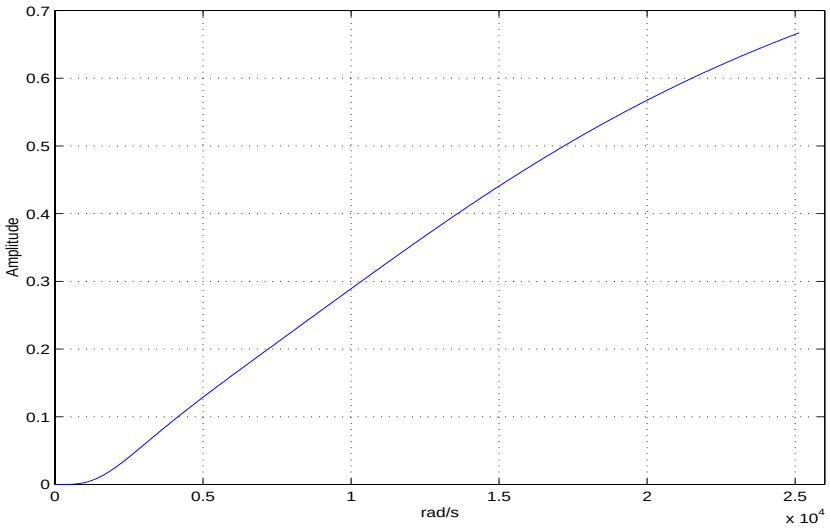

Figura 3.9: Curva da sensibilidade auditiva humana.

Finalmente para concluir a parte perceptual dos coeficientes PLP, realiza-se a compressão cúbica da amplitude, que tem por objetivo relacionar a não linearidade entre a intensidade do som real com a percebida,

$$\Phi(\Omega) = \Xi(\Omega)^{0.33} \tag{3.34}$$

Usando a equação

$$\omega = 1200\pi \sinh(\Omega/6) \tag{3.35}$$

para relacionar ω com Ω, $\Phi(\Omega)$ será representado por $\Phi(\omega)$ que será aproximado pelo espectro de um modelo só pólos usando o método da autocorrelação [2]. Este procedimento consiste em se realizar a IDFT (tipicamente de 34 pontos) em $\Phi(\omega)$ para chegar aos valores de autocorrelação e com tais valores obter os coeficientes do modelo autoregressivo de ordem M, resolvendo as equações de Yule-Walker para tal modelo.

Trabalhos utilizando coeficientes PLP mostram que eles podem ser bastante eficientes em sistemas de reconhecimento. Utilizando um número bem menor de coeficientes para tarefas independentes do locutor [27], se obtém percentuais de acerto no reconhecimento equivalentes aos obtidos pelos coeficientes mel-cepestrais. Tipicamente são usados 5 coeficientes PLP para se obter resultados equivalentes à utilização de 12 coeficientes mel-cepestrais. Para tarefas dependentes do locutor são usados tipicamente 14 coeficientes PLP [2, 27, 28, 23].

3.6.1 Algoritmo

No algoritmo para a obtenção dos coeficientes PLP, os cálculos são realizados a partir do sinal $s(n)$ segmentado em janelas de tamanho N, onde x é um segmento de $s(n)$.

i) Dado o segmento $x = [x(1)x(2)\ldots x(N)]$, calcular o espectro de potência $P(\omega) = |X(\omega)|^2$ onde $X(\omega)$ é a DTFT de x. Obter também $\Psi(\Omega)$ pela equação (3.29) e as freqüências centrais Ω_i dos filtros cb, de acordo com o número de filtros utilizados.

ii) Calcular $P(\Omega)$, $\Theta(\Omega_i)$, $\Xi(\Omega)$, $\Phi(\Omega)$ e $\Phi(\omega)$ pelas equações (3.28), (3.30), (3.31), (3.34) e (3.35), respectivamente.

iii) Calcular os M coeficientes PLP obtidos da solução do modelo autoregressivo realizada pelas equações de Yule-Walker [2], a partir da IFFT (transformada inversa de Fourier) de $\Phi(\omega)$.

3.6.2 RASTA-PLP

Como já fora mencionado, a representação do sinal de voz está relacionada com o formato do trato vocal, refletindo então seus movimentos. A taxa de variação dos componentes não lingüísticos de um sinal de fala geralmente aparece fora da taxa de variação do formato do trato vocal. Observando esta característica, foi desenvolvida a técnica RASTA (*RelAtive SpecTrAl*) que tem por objetivo tirar vantagem de

tal característica suprimindo componentes espectrais que variam mais lenta ou mais rapidamente que a taxa típica de mudança da fala. Na referência [26] é demonstrado que os coeficientes RASTA-PLP melhoram a performance de um reconhecedor na presença de ruído convolucional e aditivo.

Na prática realiza-se a filtragem de cada banda do espectro obtida no PLP antes de se calcular a equação (3.30), e então, dá-se continuidade ao procedimento igual ao PLP. Esta filtragem tem por objetivo suprimir fatores constantes em cada componente espectral de curto termo, e deste modo, tornar a representação do segmento de voz menos sensível as lentas variações do sinal de fala. Este filtro é descrito por

$$H(z) = 0.1z^4 \frac{2 + z^{-1} - z^{-3} - 2z^{-4}}{1 - 0.98z - 1} \tag{3.36}$$

3.7 Outros parâmetros

Geralmente são utilizadas outras medidas juntamente com os coeficientes anteriormente apresentados para se representar um segmento de sinal de voz. Estes outros parâmetros têm por objetivo fornecer mais informação sobre o segmento de voz e assim melhor representá-lo. As medidas mais utilizadas são: energia e delta coeficientes, sendo este último uma medida do comportamento dinâmico dos coeficientes e podendo ser calculado inclusive para energia.

O cálculo da energia foi apresentado na equação (2.2). Já os delta coeficientes analisam a variação dos coeficientes ao redor de determinado coeficiente, sendo dados por

$$\Delta c(i) = f\left(c(i-\delta), \ldots, c(i+\delta)\right) \tag{3.37}$$

para $i = 1, 2, \ldots, M$, onde M é o número de coeficientes, δ é o deslocamento a ser considerado no cálculo dos $\Delta c(i)$ e $c(i)$ são os coeficientes utilizados.

3.8 Conclusão

Neste capítulo foram apresentados os algoritmos de extração de coeficientes mais comumente observados na literatura. Através da análise teórica é possível notar a evolução nos algoritmos. Inicialmente os primeiros algoritmos, por exemplo, LPC, não inseriam nenhuma restrição quanto à percepção auditiva no cálculo dos coeficientes. Porém com o desenvolvimento de pesquisas observou-se uma grande melhoria com utilização destas restrições.

Os algoritmos apresentados seguem uma ordem cronológica, onde os últimos representam as técnicas mais novas de extração de coeficientes, por exemplo, RASTA-PLP. Os resultados referentes aos algoritmos de extração de coeficientes quando aplicados a sistemas práticos de reconhecimento são encontrados nos Capítulos 4, 5 e 6.

Capítulo 4

Análise comparativa usando ajuste temporal dinâmico

4.1 Introdução

A técnica de ajuste temporal dinâmico (*dynamic time warping*, DTW), amplamente utilizada na década de 1980 em reconhecimento de voz, é baseada na comparação de dois sinais, não necessariamente alinhados ou de mesmo tamanho, resultando em um valor de distância/similaridade entre eles.

Tipicamente, o DTW é aplicado em sistemas de reconhecimento de palavras isoladas dependente do orador com pequeno vocabulário. De fato, DTW não é normalmente utilizado em sistemas de reconhecimento que envolvam tarefas mais complexas, como por exemplo, em reconhecimento de fala contínua ou com grande vocabulário, por ser uma técnica determinística que exige um esforço computacional muito grande.

O nome DTW surgiu pelo fato de se utilizar como ferramenta a programação dinâmica na escolha do caminho ótimo e por se alinhar no tempo, "esticando" ou "comprimindo", os sinais comparados pelo algoritmo.

Este capítulo é organizado da seguinte forma: Na Seção 4.2 onde é apresentada toda a teoria necessária para compreender e implementar o algoritmo DTW. Na Seção 4.3 é apresentado o sistema de reconhecimento implementado usando DTW

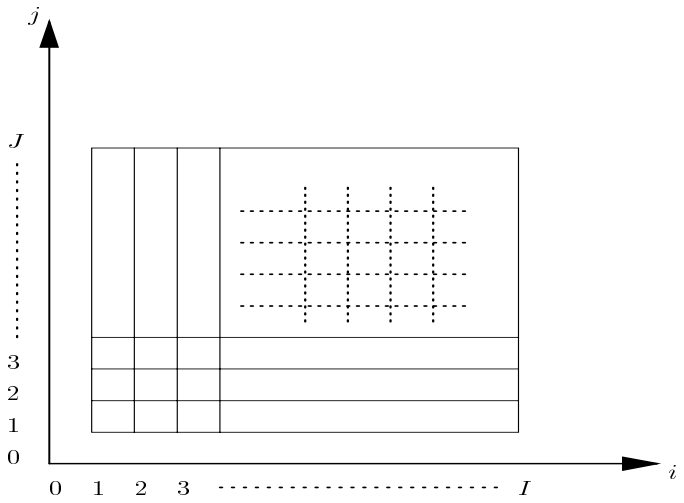

Figura 4.1: Gráfico para o estudo da programação dinâmica do algoritmo DTW.

e suas características operacionais. Na Seção 4.3.1 são descritos os procedimentos utilizados nos testes de performance de um sistema DTW por nós implementado. Estes testes utilizam os algoritmos de recorte e extração de parâmetros descritos nos Capítulos 2 e 3.

4.2 Algoritmo DTW

4.2.1 Programação dinâmica

A programação dinâmica é utilizada para analisar processos que envolvam decisões ótimas. Por este motivo este conceito é amplamente utilizado em reconhecimento de sinais de fala.

Considerando como região de aplicação deste procedimento o gráfico da Figura 4.1, onde o eixo j representa os segmentos do sinal de entrada, sendo J o número total de segmentos do sinal de entrada, e o eixo i representa os segmentos do sinal de referência, sendo I número total de segmentos do sinal de referência.

A programação dinâmica possibilita a obtenção do caminho de menor "custo" através dos pontos deste gráfico simplesmente limitando os pontos extremos, por exemplo, o caminho de menor menor "custo" indo do ponto $(1, 1)$ para o ponto

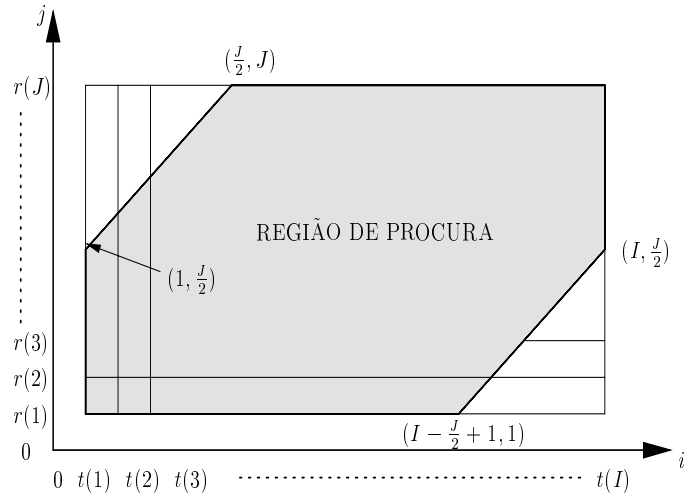

Figura 4.2: Representação paramétrica dos segmentos do sinal de entrada e da referência, $t(i)$ e $r(j)$ associados ao gráfico da restrição global de caminho.

(I, J), passando pelos pontos intermediários do gráfico.

4.2.2 Restrições globais e locais

Nesta seção serão abordados os aspectos mais práticos do algoritmo DTW relacionado ao reconhecimento de palavras faladas isoladamente.

Observam-se na Figura 4.2 os elementos $t(i)$ e $r(j)$, que são as representações dos parâmetros extraídos dos segmentos i e j, dos sinais de entrada (teste) e de referência, respectivamente. Estes elementos $t(i)$ e $r(j)$ denotam um conjunto de coeficientes paramétricos extraídos dos segmentos i e j dos sinais de fala entrada e de referência. Além disso, são comumente usados para o cálculo do custo d_N a distância de Itakura, se os vetores de parâmetros $t(i)$ e $r(j)$ forem obtidos através de coeficientes LPC, ou pela distância euclidiana ponderada, caso os parâmetros sejam dados por coeficientes cepestrais [3].

Como pode ser observado, a Figura 4.2 também apresenta uma restrição global de caminho, que tem por finalidade diminuir a região total de procura deste modo também diminuindo o custo computacional. Existem várias formas de se limitar globalmente o caminho, como por exemplo a limitação de Itakura mencionada

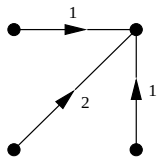

Figura 4.3: Restrição local de caminho, onde os valores indicam os pesos relativos aos vetores.

em [3]. Porém esta limitação não permite que um dos sinais comparados seja duas vezes maior que o outro. Neste trabalho foi implementada uma restrição global que encontra o sinal de menor tamanho e retira tanto do extremo esquerdo superior quanto do extremo direito inferior da região total de procura duas regiões cujas áreas quando somadas equivalem à área de um quadrado de lado igual à metade do número de segmentos do sinal de menor tamanho, como é mostrado na Figura 4.2. Isto é, na prática, se $I > J$ então a região de procura será limitada por duas retas, a primeira que liga o ponto $(1, J/2)$ ao $(J/2, J)$ e a segunda que liga o ponto $(I - \frac{J}{2} + 1, 1)$ ao $(I, J/2)$.

Na implementação prática do algoritmo DTW, deve-se ainda definir a restrição local de caminho. Como na restrição global, também existem várias opções para se limitar localmente o caminho [3]. Na Figura 4.3 é apresentada a restrição local utilizada neste projeto e que é uma das mais simples computacionalmente. Supondo que os três vetores da figura estejam convergindo para o mesmo ponto (i_k, j_k), então a distância referente a este caminho parcial é dada por

$$\min \begin{cases} D(i_{k-1}, j_k) + d(i_k, j_k) \\ D(i_{k-1}, j_{k-1}) + 2d(i_k, j_k) \\ D(i_k, j_{k-1}) + d(i_k, j_k) \end{cases} \tag{4.1}$$

onde $D(i_{k-1}, j_k)$ é a menor distância do ponto $(1, 1)$ até o ponto (i_{k-1}, j_k) e $d(i_k, j_k)$ é a distância associada ao ponto, ou seja, a distância associada a comparação de similaridade dos parâmetros correspondentes aos segmentos i_k e j_k.

Outras técnicas e definições das restrições global e local para o algoritmo DTW podem ser encontradas em [15, 16].

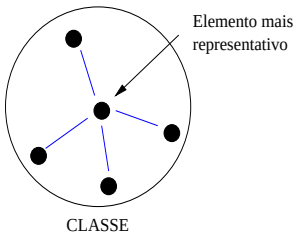

Figura 4.4: Exemplo de elemento mais representativo de uma determinada classe.

Com as restrições globais e locais de caminho definidas, deve-se definir ainda as etapas de treinamento e de reconhecimento, apresentadas a seguir.

4.2.3 Treinamento

O algoritmo DTW realiza a comparação entre dois sinais, onde um é o sinal entrada e o outro é o sinal de referência ou padrão. Cada palavra do vocabulário usado no sistema de reconhecimento representa uma classe, e o intuito do treinamento é encontrar o padrão de cada classe, podendo este ser representado por mais de um sinal, ou seja, podemos ter mais de um padrão por classe. Por exemplo, usar 3 ou 5 elementos de cada classe para treinamento significa que, para o caso de dígitos falados em português ("0", "1", ... ," 9") teremos um total de 30 ou 50 sinais. Pode-se então utilizar todos os 30 ou 50 sinais como padrões, e a classe do sinal que tiver menor D_T em comparação ao sinal de entrada será a classe reconhecida resultante. Caso o intuito seja usar somente um padrão de cada classe, seria necessário encontrar entre os 3 ou 5 elementos de cada classe aquele que melhor representa a classe. Para se encontrar este elemento que melhor representa a classe, deve-se usar o próprio algoritmo DTW que será aplicado entre todos os elementos da classe e aquele que tiver o menor somatório das distâncias em relação a todos os outros elementos da classe, será o escolhido como padrão. Se fosse possível representar os sinais por meio de somente dois coeficientes, este elemento mais representativo seria o elemento mais central da classe, como mostra a Figura 4.4.

Tendo o padrão de cada classe, facilmente aplica-se o DTW entre estes padrões e o sinal de entrada, de tal modo que, o resultado do reconhecimento será

dado pelo padrão que tiver menor distância total D_T em relação ao sinal de entrada.

4.2.4 Reconhecimento

A partir dos conceitos explicitados anteriormente sobre DTW, como as restrições global e local de procura e os próprios conceitos elementares de custo, poderemos aplicá-los ao reconhecimento de palavras faladas isoladamente usando como base um sinal padrão de classe a ser reconhecida.

Para realizarmos o processo de classificação, utilizamos os parâmetros extraídos de cada segmento tanto do sinal de entrada quanto dos padrões para se fazer a comparação entre estes sinais. Assim obtemos um valor numérico que expressa a distância entre o sinal de teste e o(s) padrão(ões). Quanto menor for este valor significa que mais "parecidos" são os sinais comparados. O critério de escolha do sinal reconhecido é dado pela classe do sinal padrão que gera o menor valor de distância quando comparado com o sinal de entrada.

Na representação do sinal de entrada no eixo j e o padrão no eixo i, então calcula-se a distância d_N como sendo a distância euclidiana ou de Itakura, dependendo do tipo de parâmetro utilizado. Exemplificando, $d_N(2,3)$ representa a distância entre os parâmetros que representam respectivamente o 2^o e o 3^o segmento do sinal de entrada e do padrão.

Na procura do caminho ótimo do algoritmo DTW é usada como base uma matriz de distância entre todos os segmentos. Inicia-se, então, a procura a partir do ponto $(1,1)$, cuja definição de distância é dada por $D(1,1) = 0$. Em seguida, dentro da primeira coluna encontram-se as distâncias totais entre o ponto $(1,1)$ e os demais pontos dentro desta coluna que fazem parte da região de procura. Prosseguindo, será feito o mesmo para a segunda coluna e assim por diante até que se chegue ao ponto (I,J) que é o ponto final da procura.

Com este desenvolvimento, $D(I,J)$ expressa o menor custo total para chegar ao ponto (I,J) a partir do ponto $(1,1)$, sendo o cálculo das distâncias totais para cada ponto da matriz de procura DTW obedece ao princípio de otimização de Bellman [3]. O resultado total da similaridade entre os sinais será dado pela seguinte

equação:

$$D_T = \frac{D(I,J)}{M} \tag{4.2}$$

onde M é o número de pontos do caminho percorrido, ou seja, quantos pontos foram necessários para se chegar de $(1,1)$ a (I,J) e D_T é a distância total normalizada. Os pontos que formam o caminho ótimo podem ser facilmente encontrados fazendo-se o caminho inverso de procura, a partir do ponto (I,J) até ao ponto $(1,1)$.

4.3 Simulações computacionais

No sistema aqui implementado, o universo de teste e treinamento são compostos respectivamente por 500 e 50 sinais, tendo o primeiro 50 sinais de cada classe e o segundo 5 de cada classe onde cada classe está associada a um dígito. Assim, haverá 50 (*teste*) ou 5 (*treinamento*) sinais que representam o dígito "1", mais 50 ou 5 sinais que representam o dígito "2" e assim por diante. Todos os sinais foram gravados em mono, 16 bits e com taxa de amostragem de 8000 Hz. Para isto, usou-se o programa Soundo'LE, que faz parte do kit Multimídia SoundBlaster da Creative e foram utilizados dois microfones comuns diferentes, um que acompanha o kit multimídia e o outro cujo modelo é Leadership com pedestal. Os sinais foram gravados em dias diferentes e com grande diversidade nos tipos de ruído ambiente.

Na segmentação do sinal voz foi utilizada janela de Hamming de 20 ms com 50% de superposição [29, 30]. Estes parâmetros estão entre os mais comumente encontrados na literatura.

Na extração de parâmetros são obtidos coeficientes LPC12, Cepestral15, LPC-Cepestral15, Mel15 (Deller et al.), Mel15 (Rabiner et al.), LPC-MEL15, PLP15 e PLP5, além da energia, delta energia e dos delta coeficientes para cada segmento. As siglas acima foram criadas para descrever os tipos e número de coeficientes utilizados no sistema de reconhecimento implementado, e estes são definidos por:

- LPC12: são extraídos 12 coeficientes LPC de cada segmento mais um coeficiente de energia. No final, cada segmento do sinal de voz será representado por 13 parâmetros.

- Cepestral15: são extraídos 15 coeficientes cepestrais de cada segmento usando a própria definição de coeficientes cepestrais além da energia do segmento. A representação final dos segmentos será dada por 32 parâmetros que incluem 1 coeficiente de energia, 15 coeficientes cepestrais, 1 coeficiente delta energia e mais 15 coeficientes delta cepestrais. No cálculo destes coeficientes foi utilizada uma FFT de 256 pontos.

- LPC-Cepestral15: o número total de parâmetros também é 32, sendo que, a única diferença é a extração de coeficientes, pois estes coeficientes cepestrais são obtidos a partir dos coeficientes LPC usando a equação (3.12).

- Mel15 (Deller et al.): são extraídos 15 coeficientes mel-cepestrais usando 20 filtros cb triangulares e FFT de 1024 pontos. O número total de parâmetros por segmento é 32, pois são usados também a energia, a delta energia e os delta coeficientes para representar o segmento [3].

- Mel15 (Rabiner et al.): a diferença deste para o anterior é o método de extração dos coeficientes mel-cepestrais [6] que é aquela expressa na Seção 3.5.2.

- LPC-MEL15: a diferença deste método para os dois anteriores é a dado pela técnica de obtenção dos coeficientes mel-cepestrais que é aquela expressa na Seção 3.5.3.

- PLP15: o número total de parâmetros continua sendo 32, e são extraídos 15 coeficientes PLP de cada segmento.

- PLP5: são extraídos 5 coeficientes PLP de cada segmento e a representação final é dada por 12 parâmetros, onde haverá 1 coeficiente de energia, 5 coeficientes PLP, 1 coeficiente delta energia e mais 5 coeficientes delta PLP.

As siglas anteriormente mencionadas são utilizadas na seção de resultados e sua implementação está diretamente relacionada com os algoritmos apresentados no capítulo 3. Nos coeficientes representados por Cepestral15, LPC-Cepestral15, Mel15 (Deller et al.), Mel15 (Rabiner et al.) e LPC-MEL15 foi realizado o procedimento

CMR apresentado na Seção 3.5.4. A utilização de coeficientes PLP com diferentes números de coeficientes tem por intuito observar a diferença comportamental em sistemas de reconhecimento que utilizam número de coeficientes característicos de sistemas independentes do locutor e que utilizam número de coeficientes característicos de sistemas dependentes do locutor.

As técnicas de reconhecimento e de treinamento utilizadas no sistema estão baseadas nas técnicas apresentadas anteriormente neste capítulo.

4.3.1 Resultados

Nesta seção serão apresentados resultados de reconhecimento relativos a testes realizados com diferentes algoritmos de extração de coeficientes, diferentes algoritmos de recorte e diferentes conjuntos de sinais de treinamento.

Na Tabela 4.1 foi utilizado um centróide para cada classe, ou seja, no algoritmo de reconhecimento DTW são realizados 10 comparações entre o sinal de teste e os 10 centróides calculados entre os 50 sinais de treinamento. Além disso, foram utilizados os diferentes coeficientes já mencionados na Seção 4.3 e os quatro algoritmos de recorte descritos no Capítulo 2, ressaltando que tanto os sinais de teste quanto os sinais de treinamento foram recortados utilizando os mesmos algoritmos de recorte. Nesta tabela, observa-se uma melhora gradativa do percentual de reconhecimento em relação ao algoritmo de recorte utilizado ao se observar os coeficientes LPC12, PLP15 e PLP5. Já entre os coeficientes cepestrais não é possível notar qualquer relação entre aumento do percentual de reconhecimento e os algoritmos de recorte. O melhor resultado absoluto é dado pelo coeficiente LPC-MEL15 usando como recorte os algoritmos 1 e 4, gerando um percentual total de acerto de 85.8%.

Na Tabela 4.2 foram utilizados 3 sinais de cada classe no treinamento, formando um conjunto de treinamento de 30 sinais. Pode-se reafirmar as observações constatadas na Tabela 4.1. Comparando as Tabelas 4.1 e 4.2, nota-se uma considerável diferença entre os percentuais de reconhecimento absoluto. De fato, usando-se 30 sinais do conjunto de treinamento como padrões de reconhecimento, apesar de aumentar o custo computacional, ocasiona uma considerável melhoria no percen-

Tabela 4.1: Percentual de acertos calculando 1 centróide entre os 5 elementos de cada classe (no treinamento).

	Algoritmos de recorte			
Coeficientes	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4
LPC12	54.4%	52.0%	57.6%	53.6%
Cepestral15	54.4%	51.2%	52.4%	61.2%
LPC-Cepestral15	51.8%	54.4%	52.0%	70.6%
Mel15 (Deller et al.)	65.8%	51.2%	43.8%	55.0%
Mel15 (Rabiner et al.)	63.6%	59.2%	68.2%	64.4%
LPC-MEL15	**85.8%**	76.0%	80.4%	**85.8%**
PLP15	38.2%	46.6%	66.4%	69.4%
PLP5	48.0%	39.4%	54.4%	64.2%

tual de acerto. Novamente o melhor resultado absoluto é apresentado pelo sistema LPC-MEL15 usando o algoritmo 4 no recorte, com um total neste caso de 93.4% de acerto.

Tabela 4.2: Percentual de acertos usando os 3 elementos de cada classe (no treinamento).

	Algoritmos de recorte			
Coeficientes	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4
LPC12	20.0%	24.8%	23.4%	27.2%
Cepestral15	70.2%	58.6%	58.0%	74.4%
LPC-Cepestral15	66.0%	60.0%	61.2%	68.4%
Mel15 (Deller et al.)	72.2%	54.8%	49.2%	62.8%
Mel15 (Rabiner et al.)	80.2%	75.4%	77.6%	79.4%
LPC-MEL15	90.8%	87.4%	87.6%	**93.4%**
PLP15	55.8%	56.4%	72.6%	77.0%
PLP5	63.8%	52.6%	63.6%	63.6%

A Tabela 4.3 mostra os resultados de forma idêntica a Tabela 4.1, exceto pelo fato de se utilizar os 5 elementos de cada classe como centróide. As mesmas observações concluídas pela análise das tabelas anteriores podem ser constatadas, e realizando uma análise comparativa, pode se notar um aumento no percentual de reconhecimento exceto pelos coeficientes LPC12 que diminuiram em mais de 15% o percentual de acerto quando comparado com os resultados obtidos na Tabela 4.1. Os melhores resultados absolutos são 93.2% e 93.0% de acerto, obtidos com os coeficientes LPC-MEL15 usando no recorte os algoritmos 1 e 4.

Tabela 4.3: Percentual de acertos usando os 5 elementos de cada classe como centróide (no treinamento).

	Algoritmos de recorte			
Coeficientes	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4
LPC12	20.4%	24.2%	23.6%	28.2%
Cepestral15	73.8%	66.0%	61.8%	75.8%
LPC-cepestral15	67.8%	67.4%	64.0%	73.8%
Mel15 (Deller et al.)	72.0%	56.4%	57.8%	57.4%
Mel15 (Rabiner et al.)	79.4%	78.6%	78.6%	83.4%
LPC-MEL15	**93.2%**	88.6%	87.4%	**93.0%**
PLP15	52.4%	59.0%	72.2%	75.0%
PLP5	59.0%	56.8%	65.2%	66.8%.

Outra constatação relativa aos coeficientes PLP pode ser observada em todas as quatro tabelas. Foi notado que na maioria dos casos o algoritmo PLP15 conseguia gerar melhor resultado que o algoritmo PLP5, ao se comparar os valores de uma mesma tabela, inclusive os melhores resultados absolutos entre este dois tipos de coeficientes foram alcançados pelo PLP15. Os únicos casos que fogem desta constatação em todas as tabelas, estão relacionados com o algoritmo 1 de recorte. Facilmente seria possível imaginar tal comportamento dos coeficientes PLP, pois os PLP15 são característicos de sistemas dependentes do locutor e os PLP5 são para sistema independentes do locutor. Uma vez que o sistema implementado era

dependente do locutor, o resultado constatado já era esperado.

As Tabelas 4.4 e 4.5 mostram as tabelas de confusão referentes aos melhores resultados absolutos encontrados nas Tabelas 4.2 e 4.3, que são respectivamente 93.0% e 93.4%.

Tabela 4.4: Tabela de Confusão - LPC-MEL15 e Algoritmo 4 da Tabela 4.2.

	Dígitos reconhecidos									
	1	2	3	4	5	6	7	8	9	0
1	50	0	0	0	0	0	0	0	0	0
2	0	50	0	0	0	0	0	0	0	0
3	0	0	36	0	0	14	0	0	0	0
4	1	0	0	49	0	0	0	0	0	0
5	3	0	0	0	46	0	0	0	0	1
6	0	0	14	0	0	36	0	0	0	0
7	0	0	0	0	0	0	50	0	0	0
8	0	2	0	0	0	0	0	48	0	0
9	0	0	0	0	0	0	0	0	50	0
0	0	0	0	0	0	0	0	0	0	50

Observando as Tabelas 4.4 e 4.5 pode se notar que os dígitos “3” e “6” são os que geram menor percentual de acerto, sendo então os maiores responsáveis pelo percentual total de acertos não ser mais alto. Esta confusão entre os dígitos “3” e “6” é devido a própria construção fonética destas palavras. Resultados semelhantes foram observados em [10, 11].

4.4 Conclusão

Neste capítulo foram estudados sistemas de reconhecimento utilizando DTW. Foram verificadas as performances dos quatro algoritmos de recorte vistos no Capítulo 2 e dos diversos tipos de características descritas no Capítulo 3 quando aplicados a

Tabela 4.5: Tabela de Confusão - LPC-MEL15 e Algoritmo 4 da Tabela 4.3.

	Dígitos reconhecidos									
	1	2	3	4	5	6	7	8	9	0
1	49	0	0	0	0	0	1	0	0	0
2	0	50	0	0	0	0	0	0	0	0
3	0	0	38	0	0	12	0	0	0	0
4	2	0	0	46	0	0	1	0	1	0
5	3	0	0	0	47	0	0	0	0	0
6	0	0	11	0	0	39	0	0	0	0
7	0	0	0	0	0	0	50	0	0	0
8	0	2	0	0	0	0	0	48	0	0
9	0	0	0	0	0	0	0	0	50	0
0	0	0	0	0	0	0	0	0	0	50

um sistema de reconhecimento de voz baseado na técnica DTW.

De modo geral foi possível constatar uma performance ligeiramente superior do algortimo de recorte 4 para quase todos os tipos de testes aqui realizados. Além disso, verificamos uma melhor representação dos sinais de voz pelos coeficientes LPC-MEL estudados na Seção 3.5.3. Estes coeficientes apresentaram performance superior até mesmo aos coeficientes PLP que estão em bastante voga em trabalhos recentes de reconhecimento de voz.

A grande desvantagem da técnica DTW perante outras técnicas usadas em reconhecimento é o custo computacional que cresce exorbitantemente em sistemas mais complexos, como aqueles independentes do orador ou mesmo para fala contínua.

Capítulo 5

Análise comparativa usando modelos escondidos de Markov

5.1 Introdução

A técnica dos modelos escondidos de Markov (*hidden Markov models*, HMM) é a mais explorada até o presente momento em reconhecimento de voz, pois ela tem se mostrado bastante eficiente na execução das mais variadas tarefas. Esta técnica surgiu com a idéia de se melhorar o modelo de reconhecimento que até então era determinístico, baseado em comparação de padrões, como é o caso do DTW.

A necessidade de uma técnica mais representativa em termos estatísticos levou os pesquisadores a desenvolver sistemas de reconhecimento usando HMM, que trata os sinais de voz como modelos estocásticos. As pesquisas com HMM em processamento de sinais de voz tiveram e ainda têm grande importância, pois foi através desta técnica que se tornou viável a implementação de sistemas inclusive para fala contínua em tempo real.

Neste capítulo será abordada a técnica de HMM para reconhecimento de sinais de voz. Na Seção 5.2 é explicado o algoritmo HMM propriamente dito. Nas Seções 5.3 e 5.4 são descritos alguns aspectos práticos do sistema, como o pré-processamento necessário e a descrição do sistema aqui implementado, respectivamente. Na Seção 5.5 são realizados os testes utilizando os algoritmos de recorte e

de extração de parâmetros.

5.2 Algoritmo HMM

Nesta seção é descrito o algoritmo HMM discreto no tempo e para tal é necessário introduzir alguns conceitos básicos de probabilidade e processo estocástico.

Observando um processo aleatório descrito como um conjunto de N estados distintos $(S_1, S_2, \ldots, S_N)$ em um instante de tempo t qualquer, de acordo com um conjunto de probabilidades associadas às transições de estado, este sistema pode ou não mudar de estado em intervalos de tempo regularmente espaçados. Se a probabilidade do estado atual q_t ser o estado S_j depende somente do estado S_i no tempo $(t-1)$, tem-se um processo markoviano de primeira ordem, que é descrito matematicamente por

$$P[q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, \ldots] = P[q_t = S_j | q_{t-1} = S_i] \tag{5.1}$$

Ao considerar esta probabilidade independente do tempo, define-se a matriz transição de estados como

$$A = \{a_{ij}\}_{N \times N} = \{P[q_t = S_j | q_{t-1} = S_i]\}_{N \times N} \tag{5.2}$$

onde a_{ij} é a probabilidade de se estar no estado S_j no instante t dado que no instante anterior $(t-1)$ se estava no estado S_i, com $1 \leq i, j \leq N$, para o caso discreto no tempo. Esta matriz de transição de estados satisfaz as seguintes propriedades

$$\begin{cases} a_{ij} \geq 0 \\ \sum_{j=1}^{N} a_{ij} = 1 \end{cases} \tag{5.3}$$

Outra notação importante é a probabilidade inicial, ou seja, a probabilidade de se começar o processo em determinado estado, denotada por

$$\pi_i = P[q_1 = S_i] \tag{5.4}$$

onde π_i é a probabilidade de se começar no estado S_i, para $1 \leq i \leq N$.

As características até então descritas definem um modelo observável de Markov, ou seja, cada estado está associado a uma saída observável do processo. Na

prática são usados modelos escondidos de Markov para modelar o reconhecimento de sinais de voz, e estes são caracterizados pela presença de dois processos sendo um observável e o outro não. Estes dois processos estão relacionados à variação das propriedades aleatórias do sinal ao longo do tempo e aos parâmetros extraídos do sinal num determinado instante de tempo. Nos HMM, a matriz de distribuição de probabilidade de símbolos de observação é dada por

$$B = \{b_j(k)\}_{N\times M} = \{P[o_t = v_k | q_t = S_j]\} \tag{5.5}$$

onde $b_j(k)$ é a probabilidade de se observar no instante t o símbolo v_k em um estado S_j, onde v_k pertence a um conjunto de M símbolos dado por $V = \{v_1, v_2, \ldots, v_M\}$, com $1 \leq k \leq M$ e o_t denota a observação no tempo t, para o caso discreto no tempo. Os elementos desta matriz atendem a propriedades equivalentes às da equação (5.3), descritas por

$$\begin{cases} b_j(k) \geq 0 \\ \sum_{k=1}^{M} b_j(k) = 1 \end{cases} \tag{5.6}$$

A definição completa de um modelo HMM depende das especificações de N, M e o conjunto de medidas de probabilidade dado pelas matrizes A, B e π. A notação usada para designar um modelo será

$$\lambda = (A, B, \pi) \tag{5.7}$$

5.2.1 Técnicas de reconhecimento

O objetivo do reconhecimento é calcular a probabilidade de determinado modelo λ gerar uma observação O determinada pelo sinal de entrada do sistema de reconhecimento, ou seja, calcular $P[\lambda|O]$. No caso do reconhecimento de dígitos isolados haverá um modelo representativo para cada dígito.

Primeiramente simplificamos o cálculo de $P[\lambda|O]$ por intermédio do teorema de Bayes através do qual pode-se concluir que maximizar $P[\lambda|O]$ é o equivalente a se maximizar $P[O|\lambda]$, ou seja, basta maximizar a probabilidade de uma seqüência dado o modelo [3, 6, 11].

5.2.1.1 Procedimento *forward*

O procedimento *forward* calcula $P[O|\lambda]$ a partir da probabilidade de ocorrer a seqüência de observações parciais $o_1, o_2, \ldots, o_t$ e o estado S_i num instante t dado o modelo λ. Esta probabilidade é definida por

$$\alpha_t(i) = P[o_1 o_2 \ldots o_t, q_t = S_i|\lambda] \tag{5.8}$$

e para se calcular $P[O|\lambda]$ utiliza-se o seguinte algoritmo:

i) Inicializando: $\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N$

ii) Recursão: $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}), \quad \begin{cases} 1 \leq t \leq T-1 \\ 1 \leq j \leq N \end{cases}$

iii) Finalizando: $P[O|\lambda] = \sum_{i=1}^{N} \alpha_T(i), \quad \begin{cases} T = \text{número de observações} \\ N = \text{número de estados} \end{cases}$

5.2.1.2 Procedimento *backward*

O procedimento *backward* calcula $P[O|\lambda]$ a partir da probabilidade de ocorrer a seqüência de observações parciais $o_{t+1}, o_{t+2}, \ldots, o_T$ dado modelo λ e que no instante t esteja no estado S_i. Esta probabilidade é definida por

$$\beta_t(i) = P[o_{t+1} o_{t+2} \ldots o_T|q_t = S_i, \lambda] \tag{5.9}$$

e $P[O|\lambda]$ dado pelo seguinte algoritmo:

i) Inicializando: $\beta_T(i) = 1, \quad 1 \leq i \leq N$

ii) Recursão: $\beta_t(i) = \left[\sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(o_{t+1})\right], \quad \begin{cases} t = T-1, T-2, \ldots, 1 \\ 1 \leq i \leq N \end{cases}$

iii) Finalizando: $P[O|\lambda] = \sum_{i=1}^{N} \pi_i b_i(o_i) \beta_1(i)$

5.2.2 Procedimento alternativo de Viterbi

O procedimento de Viterbi calcula a probabilidade $P[O|\lambda]$ a partir da seqüência de estados mais provável, e para tal define-se

$$\delta_t(i) = \max_{q_1 q_2 \ldots q_{t-1}} P[q_1 q_2 \ldots q_{t-1}, q_t = S_i, o_1 o_2 \ldots o_t|\lambda] \tag{5.10}$$

como sendo a probabilidade máxima ao longo de um único caminho, terminando no estado S_i no tempo t, considerando a seqüência de observações $o_1, o_2, \ldots, o_t$. Recursivamente pode-se calcular [6]

$$\delta_{t+1}(j) = \max_t[\delta_t(i)a_{ij}]b_j(o_{t+1}) \tag{5.11}$$

e para rastrear a seqüência de estados que maximiza a equação (5.10) (*backtracking*) utiliza-se a váriavel auxiliar $\psi_t(q_t)$ que guarda o estado no instante anterior, ou seja, $q_{t-1} = \psi_t(q_t)$.

Por questões computacionais aplica-se ao algoritmo de Viterbi o operador logaritmo, transformando as operações de multiplicação em adição. Este algoritmo é chamado de algoritmo alternativo de Viterbi sendo descrito por

i) Pré-processamento:

$$\begin{aligned}
&\overline{\pi}_i = \log(\pi_i), \quad 1 \leq i \leq N \\
&\overline{b}_i(o_t) = \log[b_i(o_t)], \quad 1 \leq i \leq N, 1 \leq t \leq T \\
&\overline{a}_{ij} = \log(a_{ij}), \quad 1 \leq i, j \leq N
\end{aligned}$$

ii) Inicializando:

$$\begin{aligned}
&\overline{\delta}_1(i) = \log(\delta(i)) = \overline{\pi}_i + \overline{b}_i(o_1), \quad 1 \leq i \leq N \\
&\psi_1(i) = 0, \quad 1 \leq i \leq N
\end{aligned}$$

iii) Recursão:

$$\begin{aligned}
&\overline{\delta}_t(j) = \log(\delta_t(j)) = \max_{\{1\leq i\leq N\}}[\overline{\delta}_{t-1}(i) + \overline{a}_{ij}] + \overline{b}_j(o_t), \quad 2 \leq t \leq T \\
&\psi_t(j) = \arg\max_{\{1\leq i\leq N\}}[\overline{\delta}_{t-1}(i) + \overline{a}_{ij}], \quad 1 \leq j \leq N
\end{aligned}$$

iv) Finalizando:

$$\begin{aligned}
&\overline{P} = \max_{\{1\leq i\leq N\}}[\overline{\delta}_T(i)] \\
&q_T = \arg\max_{\{1\leq i\leq N\}}[\overline{\delta}_T(i)]
\end{aligned}$$

v) *Backtracking*:

$$\begin{aligned}
&q_t = \psi_{t+1}(q_{t+1}) \\
&t = T-1, T-2, \ldots, 1
\end{aligned}$$

5.2.3 Técnicas de treinamento

O objetivo do treinamento é obter o modelo $\lambda = (A, B, \pi)$ satisfazendo certos critérios de otimização. Resumindo, a partir de certas seqüências de observação, se calculam as matrizes de probabilidade A, B e π.

O método de treinamento utilizado neste trabalho foi o de Baum-Welch, porém para decrevê-lo, é necessário definir algumas variáveis como

$$\begin{aligned} \xi_t(i,j) &= P[q_{t+1} = S_j, q_t = S_i | O, \lambda] = \frac{P[q_{t+1}=S_j, q_t=S_i, O|\lambda]}{P[O|\lambda]} \\ \xi_t(i,j) &= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P[O|\lambda]} \end{aligned} \tag{5.12}$$

onde $\xi_t(i,j)$ é a probabilidade de existir uma transição de S_i para S_j no instante t, considerando um modelo e uma determinada seqüência de observações. É também definida a probabilidade de estar no estado S_i em um instante t dado uma determinada observação e modelo por

$$\begin{aligned} \gamma_t(i) &= \textstyle\sum_{j=1}^{N} \xi_t(i,j), \quad t < T \\ \gamma_T(i) &= \frac{\alpha_T(i)}{P[O|\lambda]}, \qquad\quad t = T \end{aligned} \tag{5.13}$$

e a probabilidade do modelo se encontrar no estado S_i em um instante t e gerar a saída v_k é definida por

$$\delta_t(i,k) = \begin{cases} \gamma_t(i), & o_t = v_k \\ 0, & o_t \neq v_k \end{cases} \tag{5.14}$$

Considerando $\overline{\pi}_j$ como o número de vezes em que se começa no estado $S_i = \gamma_1(t)$ e H como o número de amostras de classe em questão. As matrizes A e B podem ser obtidas através das seguintes equações

$$\begin{aligned} \overline{a}_{ij} &= \frac{\text{número esperado de transições do estado } S_i \text{ para o estado } S_j}{\text{número esperado de transições a partir do estado } S_i \text{ para um estado qualquer}} \\ &= \frac{\sum_{h=1}^{H} \sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{h=1}^{H} \sum_{j=1}^{N} \sum_{t=1}^{T-1} \xi_t(i,j)} \\ &= \frac{\sum_{h=1}^{H} \sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(t_{t+1}) \beta_{t+1}(j)}{\sum_{h=1}^{H} \sum_{j=1}^{N} \sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(t_{t+1}) \beta_{t+1}(j)} \end{aligned} \tag{5.15}$$

$$\begin{aligned}\overline{b}_i(k) &= \frac{\text{número esperado de estar no estado } S_i \text{e o observar o símbolo } o_k}{\text{número esperado de vezes de estar no estado } S_i} \\ &= \frac{\sum_{h=1}^{H}\sum_{t=1}^{T}\delta_t(i,k)}{\sum_{h=1}^{H}\sum_{t=1}^{T}\gamma_t(i)} \\ &= \frac{\sum_{h=1}^{H}\sum_{t=1}^{T}\alpha_t(i)\beta_t(i)}{\sum_{h=1}^{H}\sum_{t=1}^{T}\alpha_t(i)\beta_t(i)} \qquad (5.16)\end{aligned}$$

sendo $o_t = v_k$ e o algoritmo final de Baum-Welch é caracterizado por três etapas:

i) Inicializando: Iniciar cada modelo $\lambda = (A, B, \pi)$ com valores aleatórios, atendendo as restrições (5.3) e (5.6).

ii) Recursão: Estimar os modelos através de (5.15) e (5.16). Estes modelos são reestimados após a utilização de H elementos da classe, dado pelo número de sinais por classe. Sendo a variação nos elementos das matrizes A, B e π menor que um determinado limiar, então a recursão é abortada.

iii) Repetindo: Os passos anteriores são repetidos para vários modelos iniciais com o objetivo de se estabelecer o máximo local mais favorável [3].

5.3 Pré-processamento

O pré-processamento consiste na realização da quantização dos parâmetros extraídos de cada segmento. Neste trabalho foi utilizada a quantização vetorial e através deste procedimento limitam-se os valores que os parâmetros extraídos podem assumir.

O parâmetro a ser quantizado é comparado com um dicionário (*codebook*) de parâmetros e o índice do elemento do dicionário que melhor representa o parâmetro passa a representar o próprio parâmetro. A medida utilizada para comparar um parâmetro com o dicionário no sistema implementado é a distância euclidiana.

O dicionário de parâmetros é gerado por intermédio de um conjunto de parâmetros grande o suficiente para que os elementos do dicionário possam representar todo o universo de possíveis parâmetros.

A técnica de quantização vetorial utilizada no treinamento foi a chamada de LBG (*Linde Buzo Gray*) com fissão de centróides [6] que consiste em dividir o conjunto de treinamento em 2^n elementos. Primeiramente, o dicionário é representado por um único vetor do conjunto de treinamento, e este é dado pela média dos elementos do conjunto de treino. Na segunda etapa aplica-se uma perturbação ε ao(s) elemento(s), de tal modo a podermos dividir o dicionário no dobro de regiões. Esta etapa é realizada recursivamente até que o número elementos seja atingido. Tipicamente esta perturbação varia entre $0.01 \leq \varepsilon \leq 0.05$, de acordo com [6]. Esta técnica foi utilizada somente na obtenção do vetor inicial. Posteriormente utiliza-se o LBG convencional na realização do treinamento [31].

5.4 Sistema implementado

Existem vários modelos HMM, porém o mais comumente utilizado para voz é o modelo do tipo *left-right*. Este modelo não permite transição entre quaisquer estados, ou seja, caminha-se sempre no sentido do estado mais a direita ou permanece-se parado, obedecendo às restrições temporais do sinal de voz. Como o primeiro estado é sempre o mais à esquerda do modelo, então é necessária a utilização da matriz π_i.

O modelo utilizado neste trabalho foi um modelo *left-right* com quatro estados, observado na Figura 5.1. A escolha de tais parâmetros foi decidida após a análise de [3, 6, 11, 32].

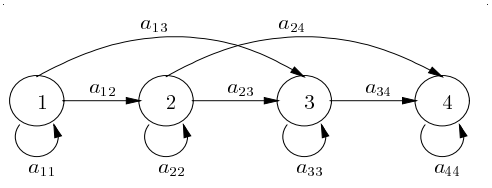

Figura 5.1: Modelo HMM do tipo *left-right* com 4 estados.

5.5 Resultados

Nesta seção são apresentados alguns resultados referentes às implementações realizadas com um sistema de reconhcimento HMM do tipo *left-right* com quatro estados. O dicionário de parâmetros de tamanho 128 se mostrou bastante eficiente, considerando a aplicação utilizada (reconhecimento de dígitos) [11, 32].

O sistema dependente do locutor utiliza a mesma base de dados aplicada no sistema DTW, já no sistema independente do locutor utilizou-se uma base de dados para o treinamento composta por 10 homens e 10 mulheres, com três repetições de cada dígito por pessoa, ou seja, 60 sinais por dígito. Estes sinais usados na base independente do locutor foram gravados a 11025 Hz em condições bastante favoráveis com pouquíssima influência do ruído ambiente. A base de teste para o sistema independente do locutor é composta por 19 homens e 19 mulheres, sendo repetido para cada pessoa três sinais por dígito, fazendo um total de 114 sinais por dígito. As pessoas utilizadas no treinamento não são as mesmas utilizadas no teste.

Tabela 5.1: Percentual de acertos de um sistema HMM dependente do orador.

	Algoritmos de recorte			
Coeficientes	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4
LPC-Cepestral15	78.2%	83.6%	87.6%	81.2%
LPC-MEL15	79.0%	80.8%	**90.6%**	**88.8%**
PLP15	61.6%	52.2%	56.6%	57.4%
PLP5	66.0%	59.4%	61.4%	67.8%

Os sinais são segmentados utilizando-se janelas de Hamming de 20 ms com 50% de superposição, e os parâmetros extraídos atendem às especificações descritas na Seção 4.3 do Capítulo 4. No sistema independente do locutor foram utilizados somente os algoritmos de recorte 1 e 2, devido às características de gravação do sinal. Uma vez que, os sinais utilizados no sistema independente do locutor não tinham suficiente informação de ruído para aplicarmos os algoritmos 3 e 4 de recorte.

As Tabelas 5.1 e 5.2, são respectivamente relacionadas a sistemas dependente

Tabela 5.2: Percentual de acertos de um sistema HMM independente do orador.

	Algoritmos de recorte	
Coeficientes	Algoritmo 1	Algoritmo 2
LPC-Cepestral15	88.0%	75.5%
LPC-MEL15	**93.2%**	87.5%
PLP15	86.8%	75.6%
PLP5	88.3%	75.4%

e independente do locutor. Comparando os resultados do classificador HMM com o classificador DTW e antecipando os resultados obtidos com o classificador ANN, podemos notar uma melhor performance do classificador HMM de um modo geral. No caso dos coeficientes LPC-MEL, o sistema usando HMM obteve resultado 90.6% de acerto para dependente do locutor contra 93.0% do DTW e 81.8% do ANN, todos usando o algoritmo 4 de recorte. Já para independente do locutor o melhor resultado do sistema usando HMM superou em aproximadamente 10% o melhor resultado usando ANN, todos com coeficientes LPC-MEL

Tabela 5.3: Variando o percentual de superposição

	Percentual de superposição			
	0%	25%	50%	75%
acerto	86.2%	**89.2%**	88.8%	84.4%
tempo(seg)	13.24	13.40	14.50	31.81

Tabela 5.4: Variando o número de coeficientes

	Número de coeficientes				
	9	11	13	15	17
acerto	84.0%	88.2%	88.6%	**88.8%**	88.6%
tempo(seg)	6.97	7.47	13.24	14.50	15.60

Tabela 5.5: Variando o tamanho do segmento

	Tamanho do segmento			
	10 ms	15 ms	20 ms	25 ms
acerto	87%	88.2%	88.8%	**89.8%**
tempo(seg)	14.60	9.78	14.50	7.58

Nas Tabelas 5.3, 5.4, 5.5, 5.6 e 5.7 são apresentados resultados relativos a "sintonia" do sistema usando classificador HMM, algoritmo 4 de recorte, dependente do locutor com janela de 20 ms, 50% de superposição, 15 coeficientes LPC-MEL15, 4 estados, *codebook* de tamanho 128 e também energia e delta coeficientes para representar os segmentos do sinal. A partir destes parâmetros, foram sendo alterados o percentual de superposição, número de coeficientes, tamanho de segmento, tamanho do *codebook* e número de estados. Sendo todas estas variações representadas nas tabelas já anteriormente mencionadas.

Tabela 5.6: Variando o tamanho do codebook

	Tamanho do codebook		
	64	128	256
acerto	88.2%	**88.8%**	87.4%
tempo(seg)	15.33	14.50	24.28

Tabela 5.7: Variando o número de estados

	Número de estados				
	3	4	5	6	7
acerto	88.8%	88.8%	88.8%	88.2%	88.8
tempo(seg)	17.30	17.41	18.12	18.70	18.62

Os melhores resultados desta "sintonia" foram obtidos com algoritmo 4 de recorte, janela de 25 ms, *codebook* de tamanho 128, 25% de superposição, 15 coe-

ficientes LPC-MEL e 4 ou 5 estados. As Tabelas de confusão 5.8 e 5.9 mostram detalhadamente os melhores resultados da "sintonia" respectivamente com 4 e 5 estados, possibilitando a observação dos dígitos mais problemáticos neste caso especificamente.

Tabela 5.8: Tabela de confusão do melhor resultado da sintonia usando 4 estados - 89.2%

	Dígitos reconhecidos									
	1	2	3	4	5	6	7	8	9	0
1	46	0	0	0	0	0	2	0	0	2
2	0	50	0	0	0	0	0	0	0	0
3	0	0	49	0	0	1	0	0	0	0
4	0	0	0	50	0	0	0	0	0	0
5	0	0	0	0	50	0	0	0	0	0
6	0	0	**46**	0	0	1	3	0	0	0
7	0	0	0	0	0	0	50	0	0	0
8	0	0	0	0	0	0	0	50	0	0
9	0	0	0	0	0	0	0	0	50	0
0	0	0	0	0	0	0	0	0	0	50

5.6 Conclusão

De um modo geral, o sistema implementado neste capítulo apresentou a melhor performance entre todos os sistemas utilizados neste trabalho.

O algoritmo 4 de recorte se mostrou mais eficiente em sistemas de reconhecimento usando HMM, assim como ocorreu no DTW, no caso independente do locutor.

Os melhores resultados do sistema usando classificador HMM foram de 90.6% de acerto para o caso dependente do locutor e 93.2% de acerto para o caso indepen-

Tabela 5.9: Tabela de confusão do melhor resultado da sintonia usando 5 estados - 90.4%

	Dígitos reconhecidos									
	1	2	3	4	5	6	7	8	9	0
1	48	0	0	0	0	0	1	0	0	1
2	0	50	0	0	0	0	0	0	0	0
3	0	0	**37**	0	0	**13**	0	0	0	0
4	0	0	0	50	0	0	0	0	0	0
5	0	0	0	0	50	0	0	0	0	0
6	0	0	**31**	0	0	**17**	0	0	0	0
7	0	0	0	0	0	0	50	0	0	0
8	0	0	0	0	0	0	0	50	0	0
9	0	0	0	0	0	0	0	0	50	0
0	0	0	0	0	0	0	0	0	0	50

dente do locutor. Comparações que serão realizadas entre os classificadores mostrarão uma melhor relação custo computacional/percentual de reconhecimento em relação as outras técnicas utilizadas neste trabalho (Capítulo 7).

Capítulo 6

Análise comparativa usando redes neurais artificiais

6.1 Introdução

As redes neurais artificiais (*artificial neural networks*, ANN) constituem um tipo de estrutura de processamento paralelo amplamente aplicado em problemas de classificação. Este tipo de estrutura tem a capacidade de "aprendizado" e por este motivo surgiu a idéia de aplicá-la em processamento de voz. No início acreditava-se que as ANN seriam a solução para os problemas onde era necessário "simular" as características perceptuais do ser humano. Porém, os resultados práticos não se mostraram tão surpreendentes, pois tal técnica não foi capaz de superar as melhores técnicas seriais utilizadas até então em tarefas de reconhecimento [33]. Atualmente as pesquisas caminham na utilização de sistemas híbridos que utilizam ANN e outras técnicas como DTW ou HMM na implementação de sistemas de reconhecimento.

Na Seção 6.2 é apresentada a teoria das ANN e os aspectos práticos relativos à implementação de um sistema de reconhecimento de dígitos isolados. Na Seção 6.3 é descrita a técnica utilizada para limitar o tamanho da entrada da rede neural. Na Seção 6.4 apresenta-se o sistema implementado neste trabalho e na Seção 6.5 é realizada a análise dos resultados obtidos com este sistema considerando os algoritmos de recorte e extração de parâmetros apresentados nos Capítulos 2 e 3.

6.2 Conceitos sobre ANN

As ANN utilizam como unidades básicas neurônios artificiais, fazendo clara analogia ao cérebro humano. Estes neurônios atuam como processadores paralelos que calculam a soma ponderada das suas entradas e aplicam a uma função limiar dada por

$$v_j = \Theta(\sum_{i=1}^{M} w_{ij} u_i) \tag{6.1}$$

onde v_j é a saída do neurônio artificial j, M é o número total de entradas do neurônio j, u_i são as entradas do neurônio, w_{ij} são os pesos de cada entrada u_i, também chamados de sinapses, e Θ denota a função limiar. Na prática, Θ pode ser implementada por várias funções do tipo degrau unitário, sigmóide, tangente hiperbólica, etc. Esta equação irá gerar um resultado que se propagará até a saída da rede, podendo esta ser discreta ou contínua. Esta idéia fica mais claramente descrita ao se observar a Figura 6.1.

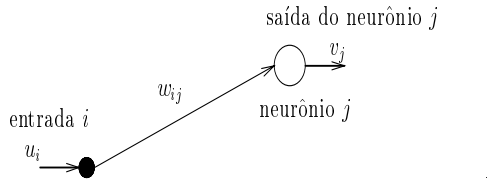

Figura 6.1: Exemplo da estrutura de um neurônio.

Existem vários tipos de redes neurais, estas são classificadas de acordo com suas estruturas. Geralmente em reconhecimento de voz são utilizadas redes do tipo MLP (*multilayer perceptron*), porém existem várias outras estruturas como por exemplo, a perceptron e a camada de Kohonen [34]. Em geral, uma rede MLP de razoável complexidade e capacidade de classificação possui três camadas, a camada de entrada, a camada escondida (intermediária) e a camada de saída, como exemplificado na Figura 6.2.

O aprendizado desta rede deve ser supervisionado, ou seja, durante o aprendizado são dadas as entradas e suas correspondentes saídas na rede. A etapa de

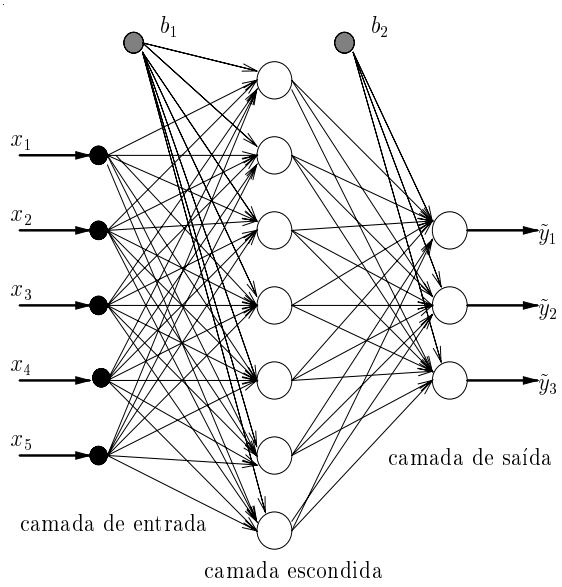

Figura 6.2: Estrutura típica de uma ANN usada em reconhecimento de voz.

aprendizado da rede é equivalente ao treinamente de um sistema de reconhecimento conforme apresentado nos Capítulos 4 e 5. O intuito desta técnica de aprendizado é conhecendo a entrada e sua respectiva saída, calcular o valor das sinapses objetivando minimizar o erro médio quadrático da saída total do sistema.

O algoritmo de treinamento mais usado na prática é o de *backpropagation* [3, 6, 15, 34]. O procedimento será descrito a partir das definições das variáveis apresentadas na Figura 6.2, onde $\underline{x} = [x_1 x_2 \dots x_m]^T$ é a entrada da rede cujo tamanho é m; $\underline{y} = [y_1 y_2 \dots y_n]^T$ é a saída ou resposta desejada da rede com relação a entrada, cujo tamanho é n; $\underline{\tilde{y}} = [\tilde{y}_1 \tilde{y}_2 \dots \tilde{y}_n]^T$ é a saída real da rede; e $(\underline{x}^k, \underline{y}^k)$ denotam os pares entrada/saída desejada, sendo $k = 1, 2, \dots, p$, onde p é o número total de pares. Quando a rede realiza p interações, é dito que ela realizou uma época. As variáveis b_1 e b_2 são chamadas de polarização (*bias*) e servem para acresecentar ao somatório das entradas ponderadas um fator constante, independente da entrada.

Dado o erro definido por

$$(\varepsilon^k)^2 = |\underline{y^k} - \underline{\tilde{y}^k}| \tag{6.2}$$

tem-se que a função objetivo dada pelo valor esperado deste erro é dada por

$$F_0(\underline{W}) = \mathcal{E}((\varepsilon^k)^2) = \frac{1}{p}\sum_{k=1}^{p}(\varepsilon^k)^2 \tag{6.3}$$

onde $\mathcal{E}$ é o operador valor esperado e, $\underline{W}$ é o vetor de sinapses da rede neural.

Em geral, a determinação do vetor ótimo $\underline{W}_0$ se dá iterativamente. Isto é, o valor do vetor $\underline{W}$ é ajustado a cada iteração k de modo que $\lim_{n\to\infty} \underline{W}_k = \underline{W}_0$.

No algoritmo *backpropagation*, o ajuste do vetor $\underline{W}_k$ é proporcional ao valor $\nabla_{\underline{W}} F_0(\underline{W})$, isto é

$$\underline{W}^{k+1} = \underline{W}^k - \alpha \nabla_{\underline{W}} F_0(\underline{W}) \tag{6.4}$$

onde α é chamado de passo do algoritmo. Calculando $\frac{\partial \mathcal{E}((\varepsilon^k)^2)}{\partial w_{ij}}$ para se obter o gradiente $\nabla_{\underline{W}} F_0(\underline{W})$, tem-se que [34, 35]

$$\frac{\partial(\varepsilon^k)^2}{\partial w_{ij}} = -2(u_i)^k(\delta_j)^k \tag{6.5}$$

sendo $(\delta_j)^k = \sum_{l=1}^{n}(\varepsilon_l)^k(\theta_{lj})^k$ o erro na saída "retropropagado" até o nó j, $(\varepsilon_l)^k = (y_l)^k - (\tilde{y}_l)^k$ o erro na saída l a ser minimizado e $(\theta_{lj})^k = \frac{\partial(\tilde{y}_l)^k}{\partial(v_j)^k}$.

Aplicando a equação (6.5) aos métodos mais comumente utilizados no treinamento das ANN, obtemos os valores de $\nabla \underline{W}^k$.

Usando o conceito de momento, a atualização dos pesos da rede se dá da seguinte forma:

$$w_{ij}^k = \beta \nabla w_{ij}^{k-1} + (1-\beta)\nabla w_{ij}^k \tag{6.6}$$

onde $\nabla w_{ij}^k = 2\alpha u_i^k \delta_j^k$ e $0 < \beta < 1$ é o coeficiente de momento.

6.3 Pré-processamento

Na seção anterior foram descritos os aspectos de uma ANN de forma bem geral. Porém quando o interesse é utilizar a ANN como um sistema de reconhecimento, é necessário limitar o tamanho da entrada da rede. Em geral as entradas da rede são os vetores de parâmetros obtidos através dos métodos de extração de

características já descritos no Capítulo 3. Já a saída vai depender exclusivamente do tamanho do vocabulário a ser reconhecido. Por exemplo, num sistema de reconhecimento de dígitos a saída da rede terá tamanho dez, correspondendo uma saída para cada dígito.

O grande problema da utilização de ANN é a necessidade do sinal de voz ser normalizado antes de ser aplicado a rede para que o tamanho da entrada seja sempre constante. Na literatura encontram-se várias técnicas para executar tal procedimento [3, 15, 23, 36, 37]. A técnica aqui escolhida para limitar o tamanho da entrada é descrita em [37], e consiste em segmentar o sinal de fala, extrair os parâmetros de cada segmento, no caso, 15 coeficientes cepestrais mais 1 coeficiente de energia, dando um total de 16 parâmetros por segmento. De acordo com [37], 100 segmentos é um número razoável para se limitar a entrada. Então são calculadas as distâncias euclidianas entre os segmentos adjacentes. Os segmentos de menores distâncias adjacentes serão eliminados enquanto o número de segmentos L for maior que 100, ou serão adicionados segmentos, simplesmente repetindo os segmentos de menor distância, enquanto L for menor que 100. Durante os procedimentos de eliminação ou adição de segmentos o valor de L deve ser atualizado.

Na prática, em nosso sistema a entrada ficou limitada em 100 segmentos com no máximo 16 parâmetros por segmentos resultando num vetor de entrada de dimensão 1600 e um vetor de saída de dimensão 10.

6.4 Sistema implementado

Nesta seção serão descritos detalhes do sistema implementado. A rede do tipo MLP utilizada possuia três camadas sendo a primeira camada de entrada, a camada escondida com 100 neurônios [37] e a camada de saída com 10 neurônios.

Como foram utilizados vários tipos de parâmetros nos teste, e alguns destes com diferentes números de coeficientes extraídos, então denotamos a dimensão da entrada pela multiplicação de 100 (segmentos) pelo número de coeficientes extraídos (m) mais um (energia), ou seja, $100(m+1)$. Os coeficientes testados são os mesmos

da Seção 4.3.

A única diferença entre a ANN implementada neste trabalho e a ANN descrita em [37] é fato da ANN usar o mesmo passo de treinamento e a mesma constante de momento β entre as diferentes camadas, o que já não ocorre na rede da referência. Na implementação foi utilizado $\alpha = 0.007$ e $\beta = 0.003$ de acordo com a referência, e o a polarização b_1 e b_2 com valor unitário.

Nos testes realizados o sistema dependente do orador foi treinado com 50 e 150 épocas, sendo que o conjunto dos sinais de treinamento é composto por cinco repetições dos sinais de cada dígito e o conjunto de teste continha 50 repetições dos sinais de cada dígito. Estes conjuntos de sinais de teste e treinamento foram os mesmos utilizados nos teste do algoritmo DTW no Capítulo 4.

No sistema independente do orador, o treinamento foi realizado com 50 e 150 épocas e com um conjunto de treinamento de 60 sinais/dígito gravados por 20 pessoas, sendo 10 do sexo masculino e 10 do sexo feminino, repetindo três vezes cada dígito. Já o conjunto de teste foi gerado por 114 sinais/dígito gravados por 38 pessoas, sendo 19 do sexo masculino e 19 do sexo feminino, repentido três vezes cada dígito. As gravações dos sinais de treinamento foram realizadas por pessoas diferentes das gravações dos sinais de teste. Devido às limitações referentes à gravação dos sinais, foram utilizados nos testes somente os algoritmos 1 e 2 de recorte, uma vez que os sinais gravados para o caso independente do locutor não contêm informação suficiente de ruído para a aplicação dos algortimos 3 e 4 de recorte, e os coeficientes LPC-cepestrais15, LPC-MEL15, PLP15 e PLP5 por uma questão de tempo e praticidade, uma vez que, na maioria dos testes realizados nesta tese, estes parâmetros têm se mostrado os mais eficientes.

6.5 Resultados

Nesta seção são realizados vários testes para se observar o desempenho da rede neural. Estes testes forma realizados com 50 e 150 passos de treinamento e com os coeficientes descritos na Seção 6.5.

As Tabelas 6.1 e 6.2 utilizam as mesmas bases de dados do sistema DTW, usadas tanto no treinamento quanto no teste. Estas tabelas só se diferenciam pelo número de passos realizados no treinamento que são respectivamente 50 e 150.

Algoritmos ineficientes de recorte podem acarretar sérios problemas na etapa de pré-processamento podendo prejudicar o sistema de reconhecimento como um todo.

Pode-se observar nas tabelas que o maior valor absoluto do percentual de reconhecimento é obtido com a utilização dos coeficientes LPC-MEL15 e do algoritmo 4 de recorte, tanto para 50 (81.8%) quanto 150 épocas (81.6%).

Tabela 6.1: Percentual de acertos para um sistema dependente do orador, treinado com 50 épocas.

	Algoritmos de recorte			
Coeficientes	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4
LPC12	54.2%	53.2%	54.6%	60.0%
Cepestral15	40.4%	23.0%	37.6%	24.4%
LPC-cepestral15	58.4%	67.6%	57.6%	60.2%
Mel15 (Deller et al.)	18.6%	15.0%	21.6%	22.4%
Mel15 (Rabiner et al.)	27.8%	23.6%	26.0%	31.4%
LPC-MEL15	65.8%	71.4%	64.0%	**81.8%**
PLP15	63.0%	62.4%	55.2%	62.2%
PLP5	40.8%	45.0%	42.4%	47.4%

Nas Tabelas 6.3 e 6.4 são realizados experimentos com as mesmas variáveis utilizadas em sistemas independentes do locutor do Capítulo 5. Pode-se observar que neste caso os coeficientes LPCMEL15 juntamente com o algoritmo 2 de recorte geram os melhores resultados absolutos.

As duas Tabelas 6.5 e 6.6 mostram com mais detalhes os resultados percentuais de reconhecimento de cada dígito, tanto para o sistema dependente quanto para o sistema independente do locutor. Por intermédio destas tabelas pode-se ter uma idéia do dígito que mais prejudicou o reconhecimento total.

Tabela 6.2: Percentual de acertos de um sistema dependente do orador, treinado com 150 épocas.

	Algoritmos de recorte			
Coeficientes	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4
LPC12	54.4%	57.2%	54.0%	64.0%
Cepestral15	40.2%	41.0%	44.4%	47.0%
LPC-cepestral15	55.8%	64.0%	58.8%	62.6%
Mel15 (Deller et al.)	18.4%	26.8%	24.6%	20.8%
Mel15 (Rabiner et al.)	23.4%	22.2%	29.4%	39.2%
LPC-MEL15	65.0%	72.6%	63.8%	**81.6%**
PLP15	62.4%	61.6%	54.0%	63.0%
PLP5	73.2%	44.8%	43.6%	49.0%

Tabela 6.3: Percentual de acertos de um sistema independente do orador, treinado com 50 épocas.

	Algoritmos de recorte	
Coeficientes	Algoritmo 1	Algoritmo 2
LPC-cepestral15	34.12%	34.58%
LPC-MEL15	77.81%	**81.06%**
PLP15	58.24%	57.98%
PLP5	58.16%	54.74%

6.6 Conclusão

Neste capítulo foi apresentada uma determinada estrutura de rede neural artificial aplicada em sistemas de reconhecimento. O objetivo foi observar o comportamento de tal sistema em tarefas de diferente complexidade computacional, variando também os parâmetros que caracterizam o sistema.

Considerando as especificações do sistema e dos sinais utilizados, os algoritmos LPC-MEL15 e o algoritmo 4 de recorte geraram os melhores resultados como

Tabela 6.4: Percentual de acertos de um sistema independente do orador, treinado com 150 épocas.

	Algoritmos de recorte	
Coeficientes	Algoritmo 1	Algoritmo 2
LPC-cepestral15	46.92%	61.15%
LPC-MEL15	79.22%	**82.71%**
PLP15	61.14%	59.29%
PLP5	59.74%	56.24%

Tabela 6.5: Percetual de acertos para cada dígito do sistema dependente do orador usando algoritmo 4 de recorte, coeficientes LPC-MEL15 e 50 épocas de treinamento.

Percentual de acertos por dígito.									
1	2	3	4	5	6	7	8	9	0
78%	88%	82%	94%	92%	36%	92%	76%	90%	90%

Tabela 6.6: Percetual de acertos para cada dígito do sistema independente do orador usando algoritmo 2 de recorte e coeficientes LPC-MEL15 e 150 épocas de treinamento.

Percentual de acertos por dígito.									
1	2	3	4	5	6	7	8	9	0
83.3%	77.2%	68.4%	91.2%	88.6%	79.8%	75.4%	84.2%	86.0%	93.0%

no Capítulo 4. Porém comparando com sistemas com o DTW e o HMM, observa-se um baixo percentual de acerto comparativamente ao da estrutura ANN aqui apresentada.

Trabalhos futuros apontam na realização de uma análise mais detalhada da rede para obtermos um conjunto de parâmetros que maximizem o percentual total de acertos. Além disto, estudos mais profundos de técnicas de pré-processamento e a implementação de sistemas híbridos de reconhecimento devem ser considerados.

Capítulo 7

Conclusão e Proposta de Trabalhos Futuros

7.1 Conclusão

Este trabalho visou estudar várias características aplicadas a sistemas de reconhecimento de voz, fazendo uma análise comparativa e atual sobre os tópicos abordados.

Inicialmente foram observados problemas relativos aos sistemas de reconhecimento de voz, sendo também apresentado um sistema genérico e simplificado de reconhecimento, cuja estrutura serviu de base para as análises realizadas.

A primeira análise realizada foi referente as técnicas de recorte. Foram descritos três algoritmos explícitos de recorte amplamente utilizados. Foi também introduzida uma nova técnica que se mostrou mais eficiente que as demais em aplicações de reconhecimento de palavras isoladas usando DTW, HMM e ANN.

Depois foi feito um estudo amplo e atual sobre algoritmos de extração de parâmetros, onde foram descritos e implementados os coeficientes LPC, cepestrais, LPC-Cesptrais, mel-cepestrais, LPC-MEL e PLP, além de energia e delta coeficientes. Observando os resultados obtidos com os classificadores DTW e ANN, notou-se o melhor desempenho dos coeficientes LPC-MEL perante os outros aqui implementados.

A análise dos classificadores foi baseada nos algoritmos DTW, HMM e ANN. Estes classificadores foram testados variando as técnicas de recorte e de extração de parâmetros utilizadas. Considerando as características utilizadas na implementação do HMM e comparando com as mesmas características de DTW e da ANN, foi possível notar a melhor performance do sistema usando HMM para o reconhecimento de palavras isoladas para a língua portuguesa falada no Brasil.

Sistemas de reconhecimento independente do locutor foram implementados usando classificadores HMM e ANN. Já os sistemas dependentes do locutor utilizaram além do HMM e ANN, a técnica do DTW. Comparando estes dois tipos de sistemas, notamos que os percentuais de acerto têm a mesma ordem de grandeza, principalmente quando comparamos os melhores resultados dos dois sistemas.

Para o caso dependente do orador, os melhores resultados obtidos para cada classificador utilizado neste trabalho foram de 93.0% para DTW, 81.8% para ANN e 90.6% para HMM. O sistema que utiliza classificador DTW foi implementado usando Matlab, o sistema ANN foi implementado em Delphi e o sistema HMM em C^{++}, sendo os tempos computacionais médios para realização dos testes com 500 sinais, respectivamente de 10319.024 segundos, 161.673 segundos e 14.560 segundos.

Já os melhores resultados para o caso independente do orador foram de 82.71% para ANN e 93.2% para HMM. Os tempos computacionais médios para a realização dos testes com 1140 sinais foram de 373.406 segundos para ANN e 34.980 segundos para HMM.

7.2 Proposta de trabalhos futuros

Em quase todas as etapas de reconhecimento pode se propor trabalhos futuros. No que se refere aos algoritmos de recorte, podem ser realizados estudos em ambientes mais problemáticos em relação ao ruído e o estudo de outras técnicas mais atuais também podem contribuir na melhoria do recorte. A implementação de algoritmos implícitos e híbridos de recorte também deve ser considerada em trabalhos futuros.

Na extração de parâmetros podem ser realizados estudos comparativos baseados nos coeficientes *mel-generalized cepstrum* e *adaptive mel-generalized cepstrum* [38, 22] com intuito de se obter uma melhor representação do sinal de voz e por conseguinte aumentar o percentual de acerto no reconhecimento. Estes algoritmos referenciados visam minimizar o erro médio quadrático na extração dos coeficientes cepestrais e mel-cepestrais.

Quanto aos classificadores, podem ser realizadas ampliações para sistemas mais complexos como por exemplo, para reconhecimento de fala contínua, limitando assim a utilização de HMM e ANN. Porém o DTW também pode ser aproveitado em sistema de reconhecimento que utilizem classificadores híbridos. A utilização de HMM contínuo, semi-contínuo e híbrido (HMM e ANN) também é um importante objeto de estudo na área de reconhecimento, até para otimizar a relação custo computacional/benefício considerando as aplicações de interesse.

Referências Bibliográficas

[1] TOKUDA, K., KOBAYASHI, T., IMAI, S., "Recursive calculation of mel-cepstrum from LP coefficients", http://kt-lab.ics.nitech.ac.jp/~tokuda/tips/mgceptr_sa2.pdf, Technical Report - Nagoya Institute of Technology - Dept. of Computer Science, April 1994.

[2] HERMANSKY, H., "Perceptual linear predictive (PLP) analysis of speech", *Journal of the Acoustical Society of America*, v. 87, n. 4, pp. 1738–1752, April 1990.

[3] DELLER, J. R. Jr., PROAKIS, J. G., HANSEN, J. H., *Discrete-Time Processing of Speech Signals*. New York, MacMillan, 1993.

[4] WILPON, J. G., RABINER, L. R., MARTIN, T. B., "An improved word-detection algorithm for telephone-quality speech incorporating both syntatic and semantic constraints", *AT&T Technical Journal*, v. 63, n. 3, pp. 479–498, March 1984.

[5] ANDRADE, M. R., dos SANTOS, S. B., MISCOW Filho, H., "Determinação de pontos terminais ('end-points') baseada no método médias-k modificado", *Anais do* XVII *do Simpósio Brasileiro de Telecomunicações*, pp. 111–114, 1999.

[6] RABINER, L. R., JUANG, B., *Fundamentals on Speech Recognition*. New Jersey, Prentice Hall, 1996.

[7] RABINER, L. R., SAMBUR, M. R., "An algorithm for determining the endpoints of isolated utterances", *Bell System Technical Journal*, v. 54, pp. 297–315, February 1975.

[8] LAMEL, L., RABINER, L. R., ROSENBERG, A., *et al.*, "An improved endpoint detector for isolated word recognition", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 29, n. 4, pp. 777–785, August 1981.

[9] WILPON, J. G., RABINER, L. R., "A modified k-means clustering algorithm for use in isolated word recognition", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 33, n. 3, pp. 587–594, June 1985.

[10] LIMA, A. A. de, *Reconhecimento de dígitos isolados usando DTW*. Projeto Final de Curso - Dep. de Eng. Eletrônica, Universidade Federal do Rio de Janeiro, Setembro 1999.

[11] FRANCISCO, M. S., *Sistema de reconhecimento de palavras isoladas em tempo real*. Projeto Final de Curso - Dep. de Eng. Eletrônica, Universidade Federal do Rio de Janeiro, Dezembro 1999.

[12] BARCELLOS, L. C., *Comunicação Pessoal*, Universidade Federal do Rio de Janeiro, Brasil.

[13] SEOK, J. W., BAE, K. S., "Endpoint detection of speech signal using discrete wavelet transform", *Proc. of the International Conference on Speech Processing*, pp. 125–129, August 1997.

[14] YING, G. S., MITCHELL, C. D., JAMIESON, L. H., "Endpoint detection of isolated utterances based on modified teager energy measurement", *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 732–735, April 1993.

[15] O'SHAUGHNESSY, D., *Speech Communications Human and Machine*. New York, IEEE Press, 2000.

[16] RABINER, L., LEVINSON, S., ROSENBERG, A., *et al.*, "Speaker-independent recognition of isolated words using clustering techniques", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 27, pp. 336–349, August 1979.

[17] de ANDRADE, M. A. R., dos SANTOS, S. C. B., "Determinação de pontos terminais (end-points) baseada no banco de filtros de coeficientes PLP", *Anais do* XVIII *Simpósio Brasileiro de Telecomunicações*, , Setembro 2000.

[18] STEVENS, S. S., VOLKMAN, J., "The relation of pitch to frequency", *American Journal of Psychology*, v. 53, pp. 329, 1940.

[19] BROOKES, M., "Voicebox: speech processing toolbox for matlab", http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html, Imperial College, London, England.

[20] FURUI, S., *Digital Speech Processing, Synthesis, and Recognition*. New York, Marcel Dekker, 1989.

[21] ROBINSON, T., "Speech analysis: lecture notes", http://svr-www.eng.cam.ac.uk/~ajr/SpeechAnalysis/index.html, University of Cambridge, Cambridge, England.

[22] FUKADA, T., TOKUDA, K., IMAI, S., "An adaptive algorithm for mel-cepstral analysis of speech", *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. I–137–I–140, March 1992.

[23] MARTINS, J. A., VIOLARO, F., "Comparison of parametric representations for hidden Markov models and multilayer perceptron recognizer", *Proc. of the International Telecommunications Society*, pp. 141–145, 1998.

[24] ATAL, B. S., "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification", *Journal of the Acoustical Society of America*, v. 55, n. 6, pp. 1304–1312, June 1974.

[25] OPPENHEIM, A. V., JOHNSON, D. H., "Discrete representation of signals", *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, v. 60, pp. 681–691, June 1972.

[26] HERMANSKY, H., MORGAN, N., "Rasta processing of speech", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 2, n. 4, pp. 587–589, October 1994.

[27] dos SANTOS, S. C. B., *Reconhecimento de voz contínua para o português utilizando modelos de Markov escondidos*. Tese de Doutorado, Pontifícia Universidade Católica do Rio de Janeiro, Dezembro 1997.

[28] dos SANTOS, S. C. B., ALCAIM, A., "Features sets in continuous speech recognition for the Portuguese language", *Proc. of the International Telecommunications Society*, pp. 126–129, 1998.

[29] ESPAIN, C., FERNANDES, R., "Isolated Digit Recognition in Konkani". Fac. de Eng. da Universidade do Pôrto and Goa University - Dep. of English.

[30] ESPAIN, C., *Comunicação Pessoal*, Universidade do Pôrto, Portugal.

[31] LINDE, Y., BUZO, A., GRAY, R. M., "An algorithm for vector quantizer design", *IEEE Transactions on Communications*, v. 28, n. 1, pp. 84–95, 1980.

[32] LIMA, A. A. de, FRANCISCO, M. S., NETTO, S. L., RESENDE, F. G. V., "Análise comparativa de parâmetros em sistemas de reconhecimento de voz", *Anais do* XVIII *Simpósio Brasileiro de Telecomunicações*, , Setembro 2000.

[33] ALCAIM, A., dos SANTOS, S. B., "Fundamentos de reconhecimento de voz", Setembro 1998. Apostila do curso ministrado pela CETUC/PUC - Rio de Janeiro, Setembro 1998.

[34] HAYKIN, S., *Neural Networks - A Comprehensive Foundation*. New Jersey, Prentice Hall, 1999.

[35] CALÔBA, L. P., SEIXAS, J. M., "Introdução as redes neurais", Julho 1999. Apostila do curso COE 724 ministrado pela COPPE/UFRJ - Rio de Janeiro.

[36] GUIMARÃES, J. G., NEGREIROS E. C., ROMARIZ, A. R., "Comparando técnicas de ajuste temporal para reconhecimento de palavras isoladas com redes

neurais", *Anais do* XV *Simpósio Brasileiro de Telecomunicações*, pp. 336–339, 1997.

[37] YNOGUTI, C. A., VIOLARO, F., "Explorando redundâncias do sinal de voz para reduzir o tempo de reconhecimento em sistemas utilizando redes neurais", *Anais do* XV *do Simpósio Brasileiro de Telecomunicações*, pp. 121–124, 1997.

[38] TOKUDA, K., KOBAYASHI, T., IMAI, S., "Adaptative cepstral analysis of speech", *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, v. 3, n. 6, pp. 481–489, November 1995.