CAPÍTULO I

INTRODUÇÃO AO RECONHECIMENTO AUTOMÁTICO DA VOZ

1.1 GENERALIDADES

O reconhecimento automático da voz (RAV) e o reconhecimento automático do locutor (RAL), têm alcançado resultados bem satisfatórios, com o crescente aumento da capacidade computacional tanto em velocidade de processamento digital quanto em memória.[1]

O RAL é executado em duas seqüências de operações: extração das características da voz do locutor, que busca obter impressões do locutor que sejam inerentes a ele, e o reconhecimento de padrões, que busca a separação entre os padrões verdadeiros e falsos. A literatura existente, registra duas formas de se fazer este reconhecimento: [1]

- **Identificação do locutor.** Procura-se dentre os N locutores aquele que se refere à locução de teste. O universo do número de decisões é do tamanho do número da população. A identificação ainda pode ser feita: com rejeição ou sem rejeição
- **Verificação do locutor.** Objetiva aceitar ou não a identidade pretensa de um locutor de teste. Dois tipos de erros podem ocorrer: falsa aceitação ou falsa rejeição. Diferentemente da identificação, neste processo existem somente duas decisões: é ou não o locutor verdadeiro. A probabilidade de um certo erro acontecer dependerá do limiar de decisão escolhido; se o mesmo for alto, uma falsa aceitação será menos provável de acontecer do que uma falsa rejeição. Qual do dois erros ocasionará um custo maior? Certamente a falsa aceitação, que permitirá o acesso de um pessoa intrusa em um ambiente de segurança, por exemplo.

No RAV consegui-se obter bons resultados quando os sinais treinados pelo modelo não são ruidosos.

Uma grande parte desta área esta voltada para o desenvolvimento de algoritmos que procuram simular atividades orgânicas. Assim, basicamente o RAV procura simular a audição humana. Contudo, existe um longo caminho que os sistemas de reconhecimento contemporâneos terão de seguir até alcançar um sistema competitivo com o da percepção humana. A distância existente entre estes dois sistemas é ocasionada basicamente pela:

- Falta de um conhecimento específico das características da voz, o que gera assim algoritmos ineficientes para o reconhecimento. Com base em vários trabalhos, pode-se dizer que a taxa de reconhecimento em certas situações não atinge resultados ótimos;[1,2,7,8,9]
- Falta de qualidade do meio de comunicação entre o homem e a máquina, agregada à inexistência de um algoritmo robusto contra o ruído, etc.

Mesmo sendo a voz complexa em sua constituição, não deixa de ser um meio natural de comunicação e uma forma de socializar os seres humanos. Devido a esta realidade, o mercado passou a exigir uma crescente " humanização" das relações entre o homem e a máquina, de modo a tornar o ambiente de trabalho mais natural. Isso, criou a necessidade de investigar as características da voz, as possibilidades de identificá-las e sua relação com a percepção da fala.[10]

O primeiro passo nessa investigação é fazer com que o computador reconheça padrões apresentados em sua entrada e gere, assim, em sua saída resultados satisfatórios. Atualmente, as pesquisas em reconhecimento têm utilizado três modelos: HMM (Hidden Markov Models - Modelos de Markov Escondidos), RNA (Rede Neural Artificial) e HMM / RNA (Modelo Híbrido) .[2]

1.2 HMM

A modelagem mais flexível e que tem conseguido um desempenho satisfatório no reconhecimento automático da voz é ainda o HMM. Tal modelo é uma máquina de estados finitos que muda de estado uma vez a cada unidade de tempo. Sua representação é baseada em uma cadeia de Markov não observável e num grupo de processos aleatórios que podem ser diretamente medidos e que representam a variabilidade acústica do fonema modelado. A principal diferença em relação à cadeia de Markov é que ao entrar em um estado, a máquina gera um vetor acústico com uma determinada probabilidade. Para que o HMM possa ser utilizado no reconhecimento da voz, são feitas suposições que são necessárias, como será visto no decorrer do compêndio.[3,12]

O HMM é definido e representado como a junção de dois processos estocásticos: a seqüência de estados do HMM, modelando a **estrutura temporal** da voz, e um conjunto de processos de saída dos estados, que modela as **características acústicas** do sinal de voz.

A estrutura temporal da voz é caracterizada pela variabilidade do tempo de pronúncia das locuções, duas ou mais repetições raramente possuem o mesmo tempo de duração, ocasionando assim um descasamento entre a locução de teste e a dos padrões armazenados.

No modelamento acústico do sinal de voz, é suposto que cada vetor acústico é descorrelacionado de seus vizinhos. Esta suposição não é completamente verdadeira, porque a inércia dos órgãos que constituem o trato vocal produz o efeito da coarticulação que assegura a correlação entre estimativas espectrais sucessivas. [1,12]

Um dos benefícios de usar este modelo no reconhecimento da voz é que ele elimina uma completa caracterização acústica de uma elocução a ser modelada. Se inicializado de forma adequada, o HMM é capaz de auto-organizar os dados acústicos em um modelo significativo e eficiente. Além do mais, os estados de um modelo tentam aproximar os fenômenos acústicos da elocução.[2]

A operação com o HMM é dividida em três fases: [1]

- **Treinamento -** O conjunto dos parâmetros acústicos do sinal de voz, também chamado de seqüência das observações, é modelado de acordo com a variação temporal da voz.
- **Reconhecimento -** No reconhecimento de palavras isoladas, a palavra a ser reconhecida deverá possuir a maior verossimilhança dentre aquelas pertencentes ao mesmo vocabulário.
- **Operação -** Após validado, o modelo está em condições para fazer o reconhecimento automático.

Apesar do HMM, no estado da arte, ser o método mais utilizado no reconhecimento da voz possui algumas deficiências, conforme será observado no decorrer deste compêndio. Isso sugere que outros métodos, ou combinações com outros métodos possam aumentar o desempenho de reconhecimento.[3]

1.3 REDE NEURAL

Inspirada na estrutura biológica do cérebro e no comportamento das células nervosas, a comunidade científica concebeu um modelo computacional a partir de elementos simples de processamento que, de forma análoga aos neurônios biológicos, realizam pequenas tarefas específicas, estão maciçamente interligados e operam em paralelo. Tais modelos, embora concepcionalmente ilimitados, encontram restrições na implementação, devido ao volume de processamento requerido e o atual estágio da tecnologia de “hardware” dos computadores.[4]

Quais as vantagens e desvantagens de um computador com relação ao cérebro humano? A capacidade aritmética computacional é bem superior mas diferentemente do cérebro, o computador não consegue se adaptar as variações provocadas por estímulos externos, isto é, não possui a capacidade de “aprender”. Para tentar suprir esta deficiência são utilizadas redes neurais artificiais.[5]

Qual a diferença entre a rede neural e os computadores convencionais? Para melhor entender a computação de uma rede neural é importante primeiro conhecer como um computador convencional, serial,

processa a informação. Um computador serial tem um processador central que pode endereçar uma linha de memória onde os dados e as instruções estão armazenadas. As computações são realizadas por um processador que lê e executa as instruções, salvando os resultados em um local específico da memória. Em um sistema serial (como também pode acontecer no paralelo) os passos computacionais são determinísticos, seqüenciais e lógicos, e o estado de uma dada variável pode ser utilizado em uma outra operação. Em comparação, as redes neurais artificiais não são necessariamente determinísticas. Não possuem um processador central complexo e sim simples unidades de processamento. Não executam instruções programadas; respondem em paralelo aos modelos de entrada apresentados. Não existem separação entre as memórias endereçáveis para armazenamento de dados. Na verdade, a informação está contida em todos os estados ativos da rede. [6]

Em que situação a rede neural poderá ser utilizada corretamente? Para capturar associações ou descobrir regularidades dentro de um grupo de modelos; onde o volume, número de variáveis ou a diversidade dos dados é muito grande; onde as relações entre as variáveis são vagamente entendidas; onde as relações são difíceis de serem descritas adequadamente com as aproximações convencionais. [6]

Quais são as suas vantagens sobre as técnicas convencionais? Dependendo da natureza da aplicação e da arrumação interna dos dados dos modelos, pode-se esperar que a rede treine melhor os dados. Isso aplica-se em problemas onde as relações são completamente dinâmicas ou não-lineares. A RNA fornece uma análise alternativa para as técnicas convencionais que são freqüentemente limitadas pela características de: normalidade, linearidade, independência de variáveis, etc. Devido ao fato da RNA capturar muitos tipos de relações, o usuário consegue modelar facilmente certos fenômenos que de outro modo poderiam ser difíceis ou impossíveis de serem modelados. [6]

Por que utilizar as redes neurais no reconhecimento automático da voz? Devido ao fato do problema de reconhecimento automático ser, basicamente, um problema de reconhecimento de padrões, razão pela qual, muitos pesquisadores as tem aplicado.[2]

Assim como o HMM, a rede neural possui deficiências. Para aumentar o grau de acerto, terão que ser feitas combinações com outros métodos de reconhecimento.[4]

1.4 MODELOS HÍBRIDOS

Para suprir as deficiências encontradas no modelo de Markov escondido contínuo e na rede neural artificial, foi criado um modelo resultante da combinação dos dois, o modelo híbrido. Neste modelo, o HMM é utilizado para o modelamento temporal da seqüência e a RNA é utilizada para introduzir no modelo a

informação contextual, que não encontra-se presente no HMM. Assim sendo, consegue-se uma melhoria na taxa de reconhecimento.[11]

1.5 OBJETIVOS DO TRABALHO

Este trabalho teve por objetivo inicial fazer uma comparação entre o desempenho da Rede Neural e do HMM, procurando encontrar as limitações destes dois modelos para uma posterior implementação do modelo Híbrido. Para realizar a comparação, foram utilizadas as palavras usadas na tese de mestrado com o título "Uso de Técnicas Neurais para o Reconhecimento de Palavras Independentes do Locutor" e na tese de graduação com o título "Comparação entre os Modelos de Markov Escondidos Comínuos e as Redes Neurais Artificiais no Reconhecimento de Voz", quais sejam: **liga**, **pare**, **grave,** **pausa**, **avance**, **siga**, **volte**, **desliga**, **ejete** e **apague**.

1.6 DIVISÃO DO COMPÊNDIO

Este compêndio é composto de sete capítulos. O Capítulo I tem por objetivo introduzir o leitor no conhecimento dos modelos que serão apresentados nos capítulos seguintes. No Capítulo II abordamos fatores relevantes do pré-processamento. No capítulo III é feita uma descrição do modelo HMM e de suas hipóteses. No capítulo IV é feita uma descrição da Rede Neural. No capítulo V são mostrados os sistemas implementados utilizando-se os três modelos. No capítulo VI são mostrados os resultados alcançados nos trabalhos de tese. O Capítulo VII apresenta a conclusão e propostas para pesquisas futuras.

CAPÍTULO II

ATRIBUTOS UTILIZADOS RAV

Neste capitulo pesquisou-se os atributos utilizados no RAV. A estimação desses atributos procura eliminar as redundâncias dos sinal, dessa forma foi utilizada menos memória, mas, sempre levando em consideração a informação mínima necessária para o reconhecimento. O discriminante de Fisher foi utilizado como ferramenta para eliminar essas redundâncias, tornando a representação mais concisa.

2.1-CONCEITOS BÁSICOS [29,30]

A Figura 2.1 mostra um modelo de sistema genérico de reconhecimento de padrões, utilizado na maioria dos sistemas de RAV (Reconhecimento Automático da Voz).

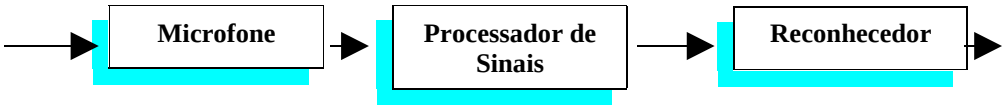


FIGURA 2.1: Reconhecimento Automático da voz

A entrada de um reconhecedor de voz é composta de um microfone e de um processador de sinais. O microfone tem a função de converter o sinal acústico em um sinal elétrico analógico e o processador de sinais analógico possui várias funções, entre elas pode-se citar:

- A conversão análogo-digital do sinal da voz.
- A estimação dos pontos terminais.
- A segmentação do sinal em janelas de tempo curto.
- A estimação dos atributos da voz.

O propósito da digitalização do sinal é produzir na saída uma representação dos dados amostrados do sinal de voz com uma relação sinal ruído (RSR) tão alta quanto possível, em reconhecimento de voz é comum utilizar um valor de aproximadamente 30dB. Após a conversão A / D, utiliza-se um filtro de pré-ênfase:

$$H_{pre} = 1 + a_{pre} z^{-1} \qquad (2.1)$$

com valores típicos para a_{pre} entre -1,0 a 4,0. A pré-ênfase proporciona um ganho no espectro do sinal de aproximadamente 20 dB / dec. Há duas razões para a sua utilização:

- As partes do sinal de voz possuem um atenuação espectral natural de aproximadamente 20 dB/ dec, devido as características fisiológicas do sistema de produção da voz.
- sistema auditivo é mais sensível as freqüências em torno de 1KHz do espectro. O filtro de pré-ênfase amplifica esta área, fornecendo ao algoritmo uma modelagem das percepções mais importantes do espectro da voz.

Depois do filtro de pré-ênfase, os pontos terminais do sinal de voz são estimados diminuindo o tempo de treinamento e logo em seguida o sinal de voz é segmentado em janelas de tempo curto onde são estimados os atributos da voz. Dentre essas funções, a que tem sido mais pesquisada ultimamente é a estimação dos atributos da voz.

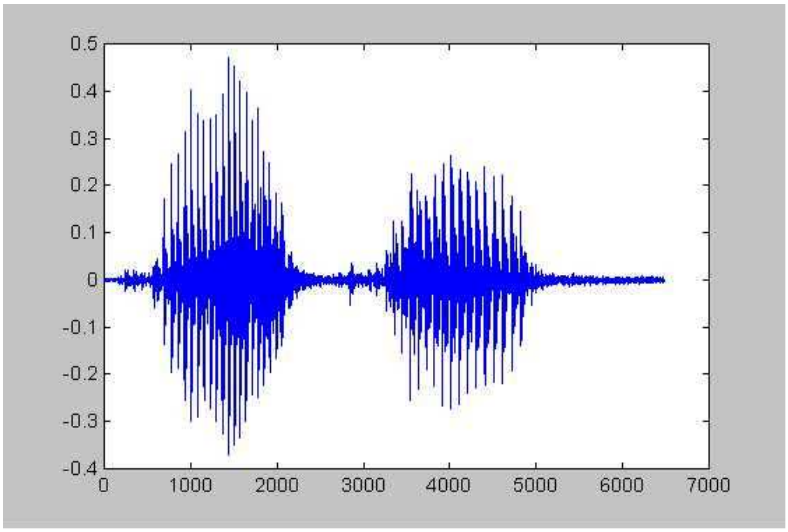


FIGURA 2.2: **Forma de onda da palavra Liga**

2.2 ESTIMAÇÃO DOS ATRIBUTOS DA VOZ

A principal suposição em muitos sistemas de processamento é que as propriedades da voz variam lentamente com o tempo. Desse modo, muitas partes da onda acústica podem ser consideradas estacionárias num intervalo que varia entre 10 e 40ms; este intervalo caracteriza o tamanho da janela a ser utilizada.[9]

2.2.1 Janelamento [31, 32, 33]

Utiliza-se janelas que possuam no domínio da freqüência, um lóbulo principal o mais estreito possível e

uma grande diferença de amplitudes entre o lóbulo principal e o primeiro lóbulo secundário, desse modo, o "fenômeno de Gibbs" ('ripple em amplitude na resposta em freqüência da janela) é amortecido.[10]

O janelamento pode ser realizado com ou sem superposição parcial entre as janelas consecutivas. A superposição aumentará a correlação entre as janelas, entretanto o tempo de processamento aumentará. No reconhecimento automático da voz, a janela de Hamming é a mais utilizada, pois proporciona maior atenuação fora da banda passante. A equação da janela de Hamming é:[2]

$$W[n] = 0.54 - 0.46.\cos(2.\pi.n / N) \qquad 0 \leq n \leq N \tag{2.2}$$

onde N , é o número de amostras.

O primeiro passo para a obtenção das características relevantes do sinal de voz é a divisão do sinal em intervalos de tempo curto, isto é, com a utilização das janelas

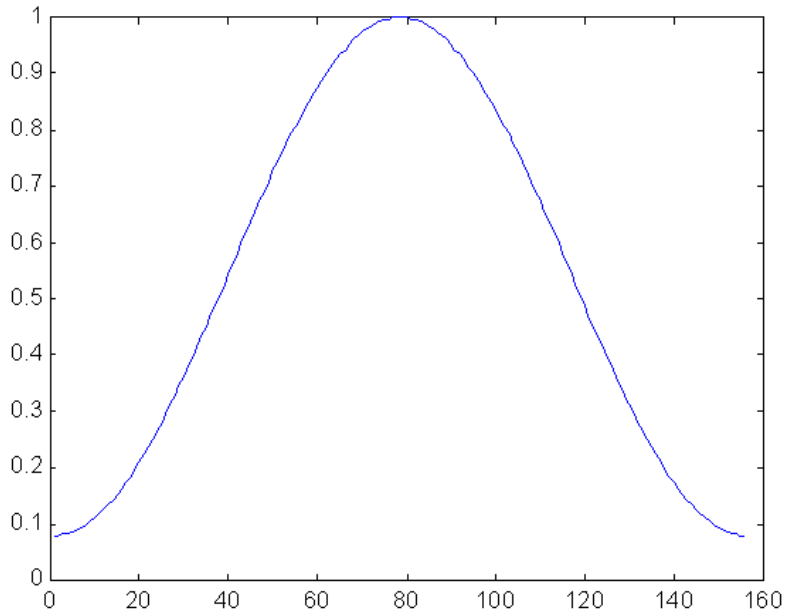


FIGURA 2.3: **Janela de Hamming com 156 amostras**

2.2.2 Atributos Utilizados no RAV[2]

Após o sinal ter sido janelado, o vetor de atributos é calculado para cada janela. Um grande número de características podem ser extraídas, para o uso no RAV, tais como: taxa de cruzamento de zeros, energia, freqüência fundamental da voz, Cepstrum, o Mel Cepstrum, etc.

sinal de voz →
- Transformada de Fourier -
  - Banco de Filtros - Energia
  - Cepestro - Coeficientes Cepestro da TF
  - Mel Cepestro - Coeficientes Mel Cepestro da TF
- Predição Linear -
  - Mel Cepestro - Coeficientes Mel Cepestro da PL
  - Cepestro - Coeficientes Cepestro da PL
  - Banco de Filtros - Energia
  - PL - Coeficientes de Predição Linear

2.2.2.1 Energia do Tempo Curto [6]

Uma das representações mais simples de um sinal é a energia. No caso de um sinal real no tempo dis-

discreto, x(n), a energia de tempo curto, em geral, é definida como:

$$E_n = \sum_{m=-\infty}^{\infty} x^2(m)h(n - m * N_f) \qquad 0 \le m \le M-1 \qquad (2.3)$$

$$0 \le n \le N-1$$

onde M representa o número de janelas, N o número de amostras e N_f é a duração de cada janela.

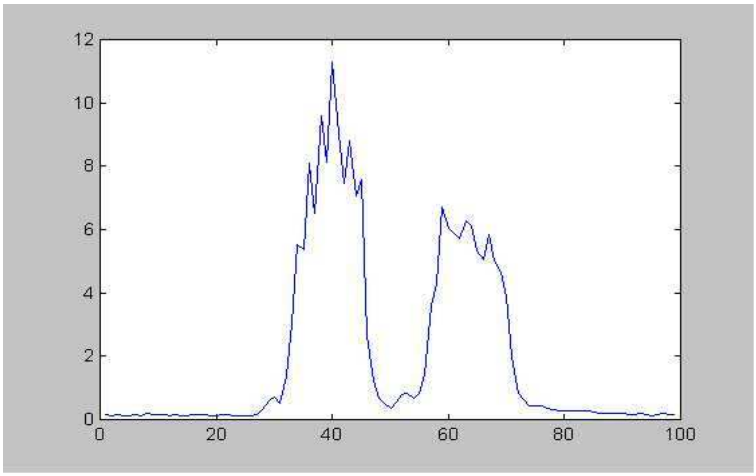


FIGURA 2.4: **Energia de tempo curto da palavra Liga**

A maior importância de E(n) é a medida de separação de segmentos vozeados dos não-vozeados, entanto, a eficiência desse atributo depende de vários fatores:

- Sensibilidade do microfone.
- Características de quantização do conversor A / D.

- Ruído presente na gravação. Isto porque, a energia em um período de silêncio pode variar consideravelmente de uma condição para outra, desta forma essas variações devem ser levadas em consideração quando o sinal de voz for é analisado.

2.2.2.2 Taxa de Cruzamento de Zeros em Tempo Curto

A taxa de cruzamento de zeros em tempo curto pode ser definida como:

$$Z_n = \sum_{m=-\infty}^{\infty} |\mathrm{sgn}[x(m)] - \mathrm{sgn}[m-1]| w(n-m) \qquad (2.4)$$

em que

$$\mathrm{sgn}[x(n)] = 1 \qquad x(n) \geq 0 \qquad (2.5)$$

$$\mathrm{sgn}[x(n)] = 1 \qquad x(n) < 0 \qquad (2.6)$$

e

$$w(n) = 1/2N \qquad 0 \leq n \leq N-1 \qquad (2.7)$$

$$= 0 \qquad \text{outros casos}$$

onde w (n - m) é uma janela retangular com N amostras.

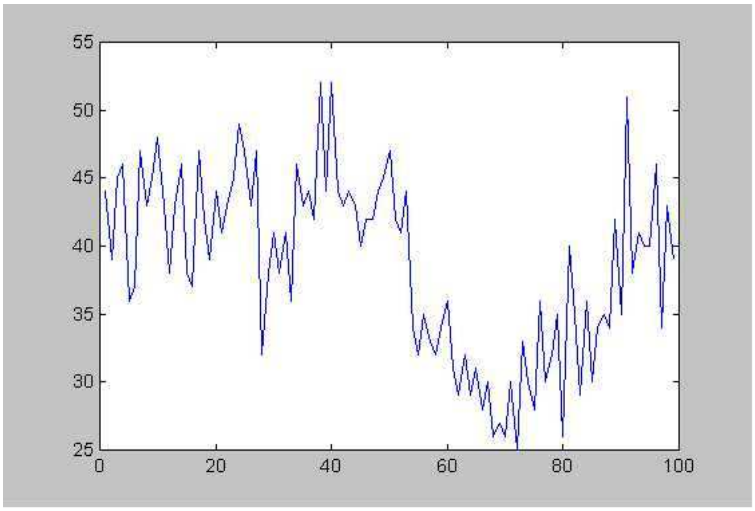


FIGURA 2.5: Taxa de cruzamento de zeros em tempo curto da palavra Liga

A taxa de cruzamento de zeros é ótima para a identificação de sons vozeados e não-vozeados. Embora o algoritmo básico para o cálculo da taxa de cruzamento de zeros requeira somente uma comparação de pares de sinais de duas amostras sucessivas; a presença do ruído na conversão A / D ou em qualquer parte do processo de digitalização, diminui a eficiência do algoritmo. [9]

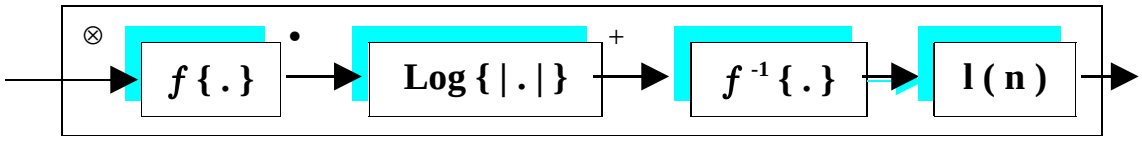
2.2.2.3 Cepstrum Real

A voz pode ser representada como a saída de um sistema linear variante no tempo e que possui propriedades que variam lentamente com o tempo. Desse modo, o principio básico da análise de voz, afirma que curtos segmentos do sinal de voz podem ser modelados como tendo sido gerados por um sistema linear invariante no tempo (LIT) excitado por um trem de pulsos quase-periódicos (segmentos vozeados) ou por um sinal de ruído aleatório (segmentos não-vozeados) .[10]

A excitação, e[n], e a resposta a impulso do trato vocal, θ[n], são combinadas por uma convolução, originando um sinal, s[n], no domínio do tempo, como mostra:

$$s[n] = e[n] \Theta \theta[n] \tag{2.8}$$

A resposta a impulso do trato vocal possui tanto a informação do locutor quanto a informação da palavra, diferentemente da excitação glotal que possui somente informações sobre as características de cada indivíduo. Esta afirmação é baseada na forma do trato vocal que é diferente entre os diversos locutores e também na locução de diferentes fonemas.[11] Desse modo, no reconhecimento da palavra isolada, a análise cepestral é utilizada para a remoção das altas freqüências do sinal , isto é, a excitação glotal.



s (n) s (w) $\hat{s}$ (w) $\hat{s}$ (n) C_θ (n)

FIGURA 2.6: **Cálculo do Cepstrum Real**

Utilizando-se a transformada de Fourier , f { . }, como um operador linear, na Equação 2.8 obtém-se

$$f\{ s[n] \} = f\{ e[n] \otimes \theta[n] \} = \tag{2.9}$$

$$= f\{ e[n] \} \bullet f\{ \theta[n] \} \tag{2.10}$$

$$= E(w) \bullet \theta(w) \tag{2.11}$$

Aplicando o logaritmo na Equação 2.11, e sabendo que o logaritmo do produto é igual a soma dos logaritmos, tem-se:

$$\hat{s}(w) = \log[s(w)] = \log \{ | E(w) \bullet \theta(w) | \} \tag{2.12}$$

$$= \log \{ | E(w) | \} + \log \{ | \theta(w) | \} \tag{2.13}$$

$$= C_e(w) + C_\theta(w) \tag{2.14}$$

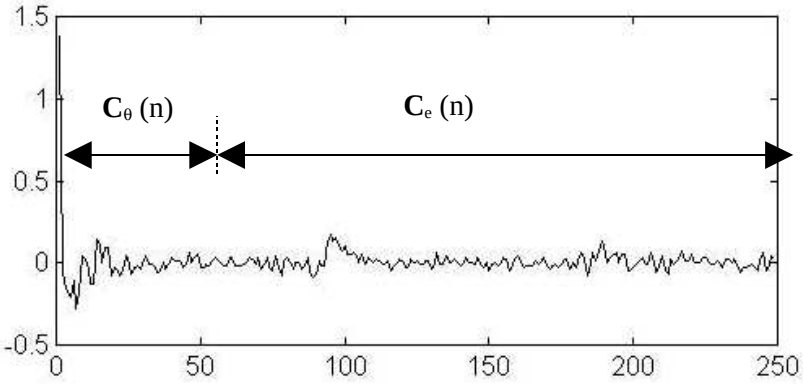


FIGURA 2.7: **Número de Coeficientes Cepstrum**

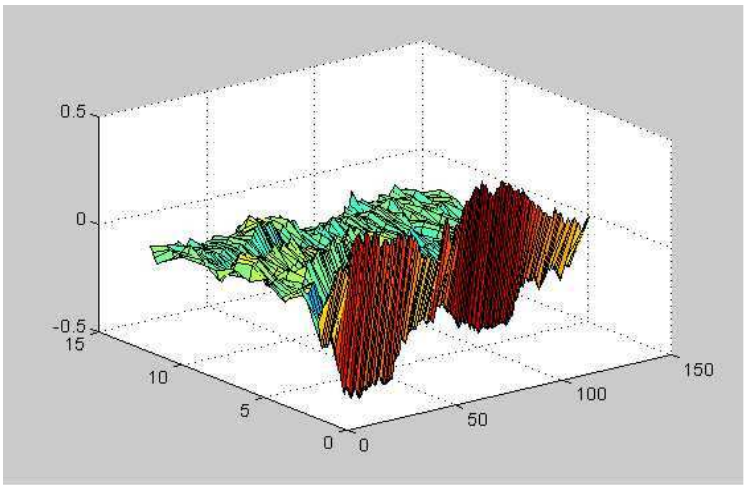
Utiliza-se a transformada inversa de Fourier na Equação 2.14, tem-se:

$$\hat{s}(n) = F^{-1}\left\{\hat{s}(w)\right\} = \tag{2.15}$$

$$= C_s(n) = C_e(n) + C_\theta(n) \tag{2.16}$$

ou seja, cepstrum representa uma operação linear no domínio da *quefrency* (anagrama da palavra *frequency*).

FIGURA 2.8: **Cepstrum real da Palavra Liga**



Com o objetivo de analisar apenas os coeficientes do trato vocal, aplica-se o processo de *liftering* (*filtering* no domínio da freqüência) para remover $C_e(n)$ de $C_s(n)$ e, neste caso, utiliza-se um *low-time lifter* (filtro passa baixas no domínio da freqüência).

2.2.2.4 Freqüências Formantes do Trato Vocal[12]

A cavidade bucal pode ser considerada um tubo de formato irregular com um número de freqüências ressonantes conhecidas como formantes. Os formantes dependem do formato e das dimensões do trato vocal, dessa forma, esse atributo traz consigo informações sobre a palavra que está sendo pronunciada e do locutor que esta pronunciando-a. Cada formante é caracterizada por uma freqüência central e uma largura de faixa. O número de formantes é variável conforme o som, entretanto a característica intrínseca das vogais que as diferenciam uma das outras parece depender principalmente das três primeiras freqüências formantes, que variam na faixa de 100 Hz a 3000 Hz e são conhecidas como: F1 , F2 e F3. Essas freqüências possuem a vantagem de não serem afetadas pelas características de freqüência do meio de transmissão, do nível de voz e da distância locutor / microfone, entretanto, a maior dificuldade com os formantes recai na sua medição. As freqüências formantes foram obtidas pelo método *Cepstrum*, algoritmo que trabalha no domínio da freqüência.

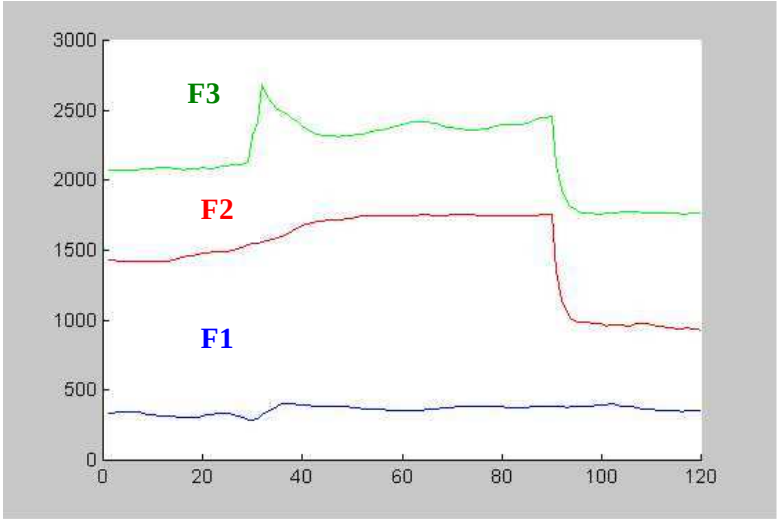


FIGURA 2.9: **Os três primeiros formantes da palavra Liga**

2.2.2.5 Faixa de Energia

Esta característica é calculada da seguinte forma: Primeiramente, o sinal de voz é dividido em quatro partes. Em cada parte do sinal são aplicados quatro filtros passa-faixas e em cada faixa é calculada a energia. A energia já foi definida pela Equação 2.3.

2.2.3 Escala Mel[2]

A escala mel baseia-se no sistema de audição humano, cuja sensibilidade aos sinais de voz se processa em uma escala não-linear de freqüências. O mel é a unidade de medida de um tom, isto é, de uma freqüência única percebida pelo ouvinte. Como referência, definiu-se a freqüência de 1 KHz, 40 dB acima do limiar de audição do ouvido, como 1000 mels. Os outros valores subjetivos foram obtidos através de experimentos onde pedia-se a ouvintes que ajustassem a freqüência física de um tom até que a freqüência percebida fosse igual a duas vezes a freqüência de referência, depois, 10 vezes a freqüência de referencia e assim por diante. Essas freqüências teriam os valores de 2000 mels, 10000 mels e assim por diante.

Para mapear a escala de freqüência acústica, f, para a escala de "percepção", f_{mel}, conhecida como escala mel deve-se seguir a relação apresentada na Equação 2.17

$$f_{mel} = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.17}$$

2.2.3.1 *Mel Cepstrum* Derivado da Transformada de *Fourier*

O *Mel Cepstrum* possui uma vantagem significativa sobre o *Cepstrum* na distinção das consoantes, uma vez que dá ênfase às altas freqüências. No *Mel Cepstrum* utiliza-se a pré-ênfase para reforçar as altas freqüências, o que tende a aumentar o erro de quantização, já que nessas freqüências o sinal é mais fraco.[2]

Alguns experimentos demonstram que a percepção humana de algumas freqüências não podem ser individualmente identificadas, dentro de certas bandas. Quando uma dessas componentes cai fora da banda, chamada de banda crítica, ela pode ser identificada. O valor dessa banda varia nominalmente de 10 a 20% da freqüência central do som, começando em torno de 100 Hz para as freqüências abaixo de 1 KHz e aumentando logaritmicamente, acima.[13]

A percepção de uma freqüência particular é influenciada pela energia dentro de uma banda critica. Logo, utiliza-se filtros de banda crítica para calcular os coeficientes *Mel Cepstrum*. Costuma-se utilizar 20 filtros passa-banda triangulares. Esses filtros espectrais são classificados como filtros de banda-crítica porque, na prática, devem estar efetivamente centrados sobre uma freqüência correspondente à freqüência de amostragem da DFT.[6] A freqüência central de cada filtro de banda crítica é calculada como:

$$F_{c,i} = K_i \frac{f_s}{N} \tag{2.18}$$

onde f_s é a freqüência de amostragem e N é o número de pontos usado no cálculo da transformada discreta de *Fourier* e k_i é o ponto da DFT correspondente à freqüência central de cada filtro. Para o primeiro filtro, X são os pontos na DFT que variam de 1 a 19:

a) a reta à esquerda da freqüência central K_i

$$F(X) = \frac{X}{K_i(i)} \qquad 1 \le X \le 10 \tag{2.19}$$

b) a reta à direita da freqüência central K_i

$$F(X) = \frac{X - K_i(2)}{K_i(i) - K_i(i-1)} \qquad 11 \le X \le 20 \tag{2.20}$$

Para os filtros seguintes:

a)reta à esquerda à esquerda da freqüência central K_i

$$F(X) = \frac{X - k_i(i)}{K_i(i) - K_i(i-1)} + 1 \tag{2.21}$$

b) reta à direita da freqüência central K_i

$$F(X) = \frac{X - K_i(i)}{K_i(i) - K_i(i+1)} + 1 \qquad (2.22)$$

onde: X são os pontos da DFT pertencentes ao filtro K_i

K_i número da DFT referente a freqüência central , $2 \leq i \leq 20$.

2.2.3.2 *Mel Cepstrum* Derivado dos Parâmetros LPC [11]

A desvantagem da utilização da técnica de *Fourier* para o cálculo dos coeficientes *Mel Cepstrum* é o tempo de processamento que é aumentado cerca de duas vezes se comparado com a análise da predição linear entretanto, em ambientes ruidosos os coeficiente calculados com a DFT são mais robustos com relação ao ruído.

A idéia principal da análise da predição linear é que a amostra atual da voz pode ser aproximada por uma combinação linear das amostras passadas. Dado um sinal s(n), define-se o modelo de predição linear como:

$$s(n) = \sum_{i=1}^{N_{LP}} a_{LP}(i)s(n-i) + e(n) \qquad (2.23)$$

onde N_{LP} representa o número de coeficientes do modelo (ordem de predição), os $\{a_{LP}\}$ os coeficientes de predição linear (coeficientes do preditor) e e(n) o erro médio quadrático (a diferença entre o valor predito e o valor medido).

Um modelo de filtro bastante utilizado, para o sinal de voz é o modelo autoregressivo. O modelo autoregressivo representa muito bem a função de transferência do trato vocal para os sons vozeados não nasalados, principalmente as vogais. Apesar da dificuldade, desse filtro, em modelar os fonemas nasalados e fricativos, como a dificuldade de se estimar o segundo formante do fonema /m/ devido ao zero próximo, os modelos autoregressivos, são os mais utilizados pelos pesquisadores, para a estimação das envoltórias espectrais, por vários motivos: [19]

- Exige pequena carga computacional, com a conseqüente economia de tempo e memória.
- Os zeros podem ser modelados por pólos, isto é, um zero em z = a (| a | < 1) pode ser exatamente representado por um número infinito de pólos

$$(1 - az^{-1}) = \frac{1}{1 - \sum_{n=1}^{\infty} \left(az^{-1}\right)^n} \qquad (2.24)$$

pois para algumas soluções (método da autocorrelação, por exemplo) o filtro é garantidamente estável e de fase mínima;

- Usualmente é desnecessário acrescentar-se mais pólos para as vogais nasais, apesar dos formantes extras em tais fonemas, pois os formantes de alta freqüência em nasais possuem banda larga e consequentemente tão pouca energia que um modelamento espectral acurado é de menor importância.
- Além disso, os zeros têm menor importância na percepção do som que os pólos, pois estes determinam as freqüências de ressonância, enquanto aqueles, na maior parte dos casos, apenas alteram a forma do espectro, ou seja, o ouvido humano percebe melhor os picos do espectro que os vales. O efeito prático dos zeros na estimativa dos formantes será de atenuar a amplitude dos pólos que caiam em sua proximidade.

Supõe-se que no processo aleatório uma entrada espectralmente invariante, torna-se possível estimar a densidade de potência espectral da saída.

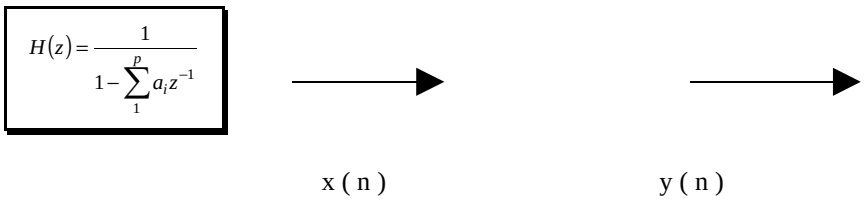


FIGURA 2.10: **Função de Transferência do Tipo "só-de-pólos"**

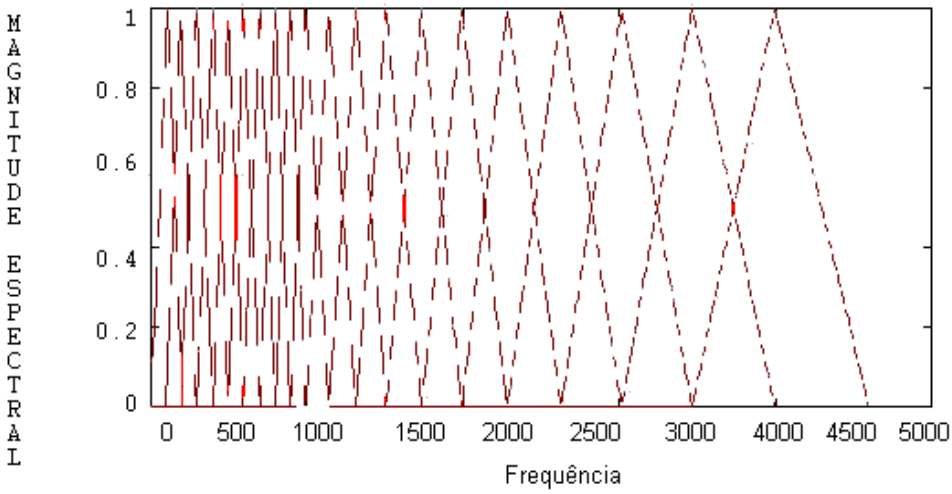


FIGURA 2.11: **Filtros Utilizados no cálculo dos coeficientes *Mel Cepstrum***

A Tabela 2.1 mostra as freqüências de corte e largura de banda crítica, de acordo, com a Equação 2.18.

TABELA 2.1: Freqüências de corte e a largura de banda crítica

Faixa Linear			Faixa Log.		
k_i	Filtro	$F_{c,i}$	k_i	Filtro	$F_{c,i}$
1	100	10	11	1148	107
2	200	19	12	1318	123
3	300	28	13	1514	141
4	400	38	14	1737	162
5	500	47	15	1995	186

6	600	56	16	2292	213
7	700	66	17	2692	245
8	800	75	18	3020	281
9	900	84	19	3467	323
10	1000	93	20	4000	372

De posse da evolução da energia do sinal obtida a partir dos filtros faz-se o cálculo dos coeficientes *Mel Cepstrum* utilizando a Equação 2.33.

$$MFCC_i = \sum_{k=1}^{K} X_k \cos\left[i\left(k - \frac{1}{2} \right) \frac{\pi}{K} \right] \tag{2.25}$$

onde i = 1, 2,..., M, sendo que M representa o número de coeficientes mel, e X_k representa o log-energia da saída do k- ésimo filtro, onde k é o número que varia de 1 até K (K=20).

A Figura 2.12 representa os 12 coeficientes *Mel Cepstrum* do comando liga.

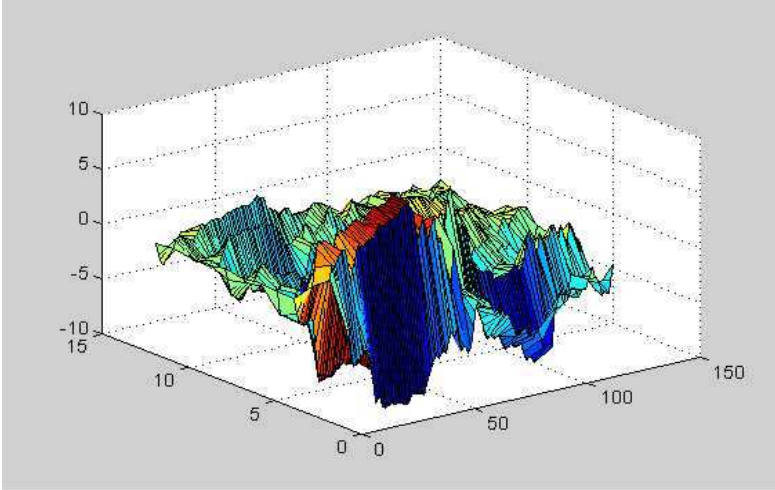


FIGURA 2.12: **Gráfico dos 12 coeficientes *Mel Cepstrum* da palavra Liga**

2.2.4 Coeficientes Delta[2]

Estes parâmetros são obtidos através das derivadas de primeira ordem de determinados atributos. São utilizados para representar as mudanças dinâmicas no espectro da voz e, desse modo, detectam variações bruscas dentro do espectro. As aproximações mais populares são :

$$s(n) \equiv \frac{d}{dt} s(n) \approx s(n) - s(n-1) \tag{2.26}$$

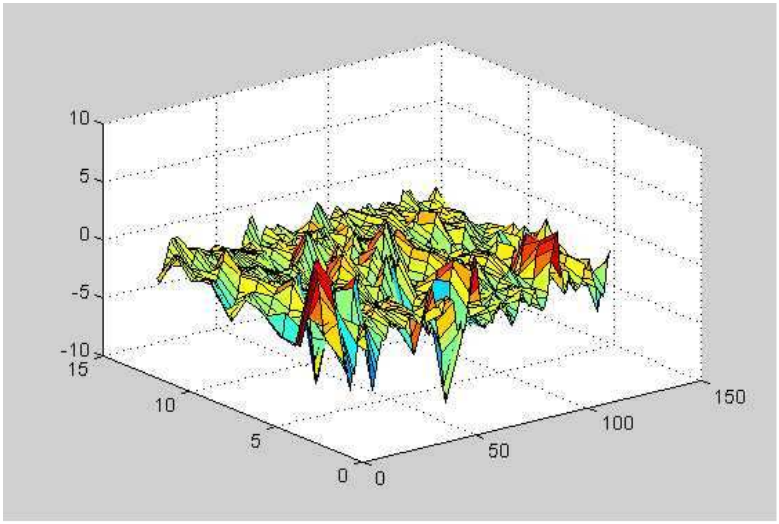


FIGURA 2.13: **Gráfico dos 12 coeficientes *Delta Mel Cepstrum* da palavra liga.**

Os parâmetros de segunda ordem são obtidos reaplicando a derivada sobre os resultados obtidos na primeira derivação.

$$\dot{s}(n) = \frac{d}{dt}s(n) \approx s(n+1) - s(n) \tag{2.27}$$

2.3 ESTUDOS DOS ATRIBUTOS UTILIZADOS NO RAV [6,14]

Um dos problemas encontrados no reconhecimento de padrões é a dimensionalidade da matriz dos atributos. A técnica utilizada, nesse trabalho, para reduzir a dimensão foi o discriminante de *Fisher*. O emprego desse determinante tem por objetivo verificar a eficiência relativa de um determinado atributo em relação aos outros atributos no reconhecimento dos dez comandos (liga, pare, grave, pausa, avance, siga, volte, ejete, desliga e apague). O discriminante de *Fisher* pode ser assim definido

$$J = \frac{S_B}{S_w} = \frac{\text{variância entre - classes}}{\text{variância intra - classes}} \tag{2.28}$$

onde as classes, são caracterizadas pelos diferentes comandos que compõem o vocabulário utilizado. O objetivo desse discriminante é encontrar atributos que possuam uma distância entre classes alta e uma distância intra-classes baixa. A Figura 2.14 mostra como a palavra liga pode ser facilmente diferenciada da palavra avance utilizando o 1º coeficiente *Mel Cepstrum*.

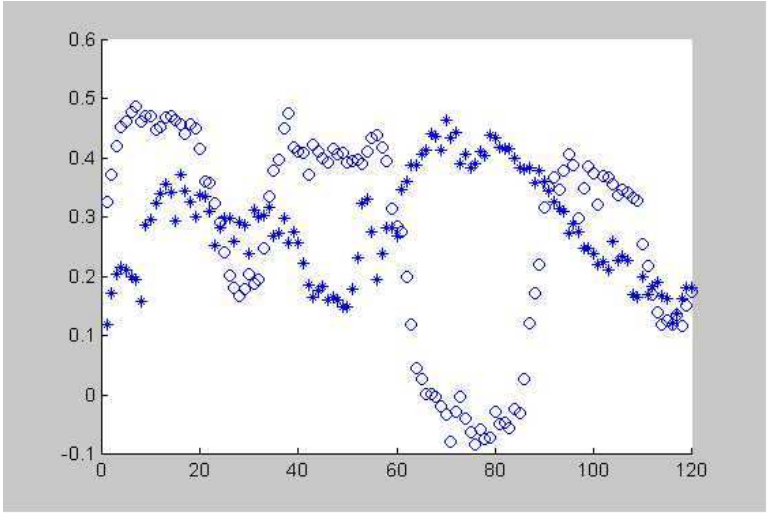


FIGURA 2.14: Com a utilização do 1º coeficiente *Mel Cepstrum*, o reconhecimento da palavra Liga, '+', e da palavra Avance, 'o', tornou-se mais fácil.

Foram analisados 55 atributos: 12 coeficientes *Mel Cepstrum*, 12 coeficientes *Cepstrum*, 12 coeficientes *Delta Mel Cepstrum*, 12 coeficientes *Delta Delta Mel Cepstrum*, 1 taxa de cruzamento de zeros normalizada, 1 log energia, 1 *Delta log* energia , 3 energias normalizadas e os formantes.

A Tabela 2.2 apresenta em ordem decrescente, os atributos que serão utilizados nesse trabalho. A Tabela 2.3 tem os atributos que foram considerados inaptos para o reconhecimento dos dez comandos.

TABELA 2.2 - Atributos utilizados no reconhecimento dos dez comandos

CARACTERÍSTICAS	DISCRIMINANTE DE FISHER
Taxa de Cruzamento de zeros Normalizada	3.7603
2º *Mel Cepstrum*	1.8083
1º *Cepstrum*	1.5802
Faixa de Energia	1.3985
3º *Mel Cepstrum*	1.0930
1ª Energia Normalizada	0.9937
1º *Mel Cepstrum*	0.9292
7º *Mel Cepstrum*	0.8279
4º *Mel Cepstrum*	0.8256
6º *Cepstrum*	0.7568
5º *Cepstrum*	0.6897
2ª Energia Normalizada	0.6310
6º *Mel cepstrum*	0.5213
4º *Cepstrum*	0.5197
2º *Cepstrum*	0.5081
3º *Cepstrum*	0.4666
5º *Mel Cepstrum*	0.4290
9º *Coeficiente Mel Cepstrum*	0.4243

TABELA 2.3: **Atributos que não foram utilizados no reconhecimento dos dez comandos**

CARACTERÍSTICA	DISCRIMINANTE DE FISHER
11º Coeficiente *Mel Cepstrum*	0.3546
10º Coeficiente *Mel Cepstrum*	0.3254
7º Coeficiente *Cepstrum*	0.3104
8º Coeficiente *Mel Cepstrum*	0.3006
11º Coeficiente *Cepstrum*	0.2969
12º Coeficiente *Mel Cepstrum*	0.2603
9º Coeficiente *Cepstrum*	0.2446
8º Coeficiente *Cepstrum*	0.2213
10º Coeficiente *Cepstrum*	0.2273
1º Coeficiente *Delta Mel Cepstrum*	0.2239
12º Coeficiente *Cepstrum*	0.2077
Log Energia	0.1499
3º Coeficiente *Delta Mel Cepstrum*	0.1488
2º Coeficiente *Delta Mel Cepstrum*	0.1480
7º Coeficiente *Delta Mel Cepstrum*	0.1445
6º Coeficiente *Delta Mel Cepstrum*	0.1440
5º Coeficiente *Delta Mel Cepstrum*	0.1240
12º Coeficiente *Delta Mel Cepstrum*	0.1153
3º Coeficiente *Delta Delta Mel Cepstrum*	0.1130
8º Coeficiente *Delta Mel Cepstrum*	0.1120
10º Coeficiente *Delta Mel Cepstrum*	0.1062
9º Coeficiente *Delta Mel Cepstrum*	0.1038
4º Coeficiente *Delta Mel Cepstrum*	0.1000
2º Coeficiente *Delta Delta Mel Cepstrum*	0.0982
4º Coeficiente *Delta Delta Mel Cepstrum*	0.0902
5º Coeficiente *Delta Delta Mel Cepstrum*	0.0874
11º Coeficiente *Delta Mel Cepstrum*	0.0859
1º Coeficiente *Delta Delta Mel Cepstrum*	0.0756
8º Coeficiente *Delta Delta Mel Cepstrum*	0.0751
7º Coeficiente *Delta Delta Mel Cepstrum*	0.0719
6º Coeficiente *Delta Delta Mel Cepstrum*	0.0660
9º Coeficiente *Delta Delta Mel Cepstrum*	0.0651
10º Coeficiente *Delta Delta Mel Cepstrum*	0.0573
12º Coeficiente *Delta Delta Mel Cepstrum*	0.0531
Delta log Energia	0.0517
11º Coeficiente Delta Delta "Mel Cepstrum"	0.0345

CAPÍTULO III

HMM

Neste capítulo foram apresentados os seguintes itens: conceitos básicos, composição e as suposições necessárias para a sua utilização do HMM. Também serão abordados: os tipos, a estrutura e os três problemas básicos encontrados no HMM. Esses problemas básicos foram resolvidos no desenvolvimento do algoritmo *Segmental K-means*. Este capítulo foi concluído, falando sobre a inclusão da densidade de duração de estado e as principais limitações encontradas no HMM para sua utilização no reconhecimento automático da voz.

3.1 CONCEITOS BÁSICOS DO HMM

O problema básico no reconhecimento da voz é habilitar na saída uma seqüência, que seje a mais provável, dado os vetores acústicos. Assim, escolhe-se uma seqüência de palavras W, de modo que a probabilidade *P(W/O)* seje maximizada; sendo *O* uma seqüência de vetores acústicos. [1]

$$P(W/O) = \frac{P(W)P(O/W)}{P(O)} \tag{3.1}$$

O primeiro termo, *P(W),* representa a probabilidade a priori de se observar *W*, independente do sinal observado, e é determinado pelo *Modelo de Linguagem.* Esse tipo de modelagem é utilizada para o reconhecimento de fala contínua e de palavras conectadas, definindo a gramática que será utilizada. O modelamento da linguagem foge dos objetivos desse trabalho e, assim, não será estudado.

Os outros dois termos, $P(O/W)$ / P (O), são determinados pelo Modelo Acústico. Como *P(O)* é igual para todos os modelos acústicos, o processo restringe-se a maximizar *P (O / W).*

Todavia, é suposto que essas modelagens sejam independentes entre si, podendo ser dessa forma estimadas separadamente.[3]

3.2 MODELAMENTO ACÚSTICO

O objetivo do modelamento acústico é proporcionar um método eficaz para o cálculo da verossimilhança de qualquer seqüência de vetores *O* dada uma palavra W_i ,ou seja, P (O / λ_i), λ_i é o modelo associado à palavra W_i. Essas seqüências de palavras são decompostas em sons básicos chamados de *Unidades Fonéticas* (fones, difones, trifones, etc) e cada uma dessas unidades é representada por *HMM*'s. O *HMM* é representado como a junção de dois processos estocásticos: a seqüência de estados do HMM, modelando a estrutura temporal da voz e um conjunto de processos de saída dos estados, modelando as características acústicas dos sinais de voz.[20, 24]

3.3 COMPOSIÇÃO DE UM HMM [7]

Seja $y \in Y$ uma variável representando as observações e i, $j \in X$ variáveis representando estados do modelo λ_i. Pode-se definir λ_i, de acordo com os itens relacionados abaixo.[1, 9]

3.3.1 Matriz de Transição entre os Estados

$$a_{ij} = P(q_{t+1} = j / q_t = i) \qquad 1 \le i, j \le N \tag{3.2}$$

é a probabilidade de transição do estado i para o estado j, onde N é o número de estados. Para se obter uma rápida convergência, a matriz de probabilidades de transições deverá conter valores aleatórios entre zero e um, respeitando-se a restrição estocástica e o avanço e retorno entre os estados.

3.3.2 Influência da Probabilidade de Transição entre os Estados do HMM

A estrutura do HMM define a evolução temporal do sinal a ser modelado. Esta estrutura é representada pela matriz *A*. Dentre as estruturas, a que melhor representa a variação temporal da voz é a chamada: "Esquerda-Direita", mas alguns autores utilizam o modelo *Ergótico*.

3.3.2.1 Modelo *Ergótico*

Não restringe nenhuma transição entre os estados, qualquer estado pode ser alcançado a partir de qualquer outro estado, isto é,

$$a_{ij} \ge 0, \text{ para todos } i \text{ e } j. \tag{3.3}$$

A principal desvantagem é a dificuldade em modelar a seqüência temporal dos eventos acústicos, além de aumentar a convergência para um máximo local. Abaixo têm-se um exemplo de uma matriz referente ao Modelo *Ergótico* com três estados.[5]

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

3.3.2.2 Modelo Esquerda-Direita [2]

O modelo Esquerda-Direita tem as seguintes regras**:**

- nenhuma transição é permitida para estados cujo índice seje menor do que o atual, isto é,

$$a_{ij} = 0 \text{ para todo } j < i. \tag{3.4}$$

- a probabilidade do estado inicial é igual a:

$$\Pi_{i=1} = 0 \text{ e } \Pi_{\forall \neq 1} = 0 \tag{3.5}$$

Freqüentemente, são impostas restrições nas transições do modelo, atribuindo-se a cada transição um número máximo de estados que pode ser alcançado, isto é,

$$a_{ij} = 0 \quad \text{para todos } j > i + \Delta \tag{3.6}$$

quando Δ for igual a 2, tem-se um modelo particular denominado Modelo de *Bakis*, modelo este utilizado nesta tese. A matriz, abaixo, é um exemplo de uma matriz de transição para o Modelo de *Bakis* com seis estados.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 & 0 \\ 0 & a_{22} & a_{23} & a_{24} & 0 & 0 \\ 0 & 0 & a_{33} & a_{34} & a_{35} & 0 \\ 0 & 0 & 0 & a_{44} & a_{45} & a_{46} \\ 0 & 0 & 0 & 0 & a_{55} & a_{56} \\ 0 & 0 & 0 & 0 & 0 & a_{66} \end{bmatrix}$$

Sendo que, $a_{N\,N=1}$

$a_{N\,i} = 0,\ i < N$

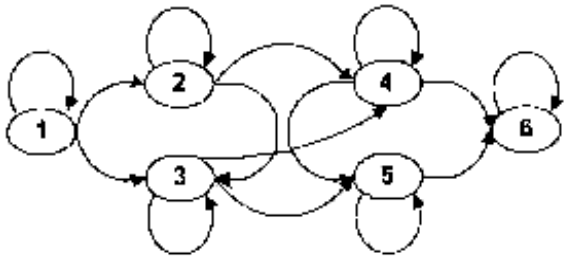


FIGURA 3.1: **Modelo de *Bakis* referente a matriz A**

3.3.3 Matriz das Probabilidades das Observações, $b_j(o)$.

A matriz das probabilidades de observações é responsável pela distribuição acústica nos estados do modelo e influencia diretamente a distribuição das observações nos estados.[11]

3.3.3.1 Influência da Distribuição das Probabilidades das Observações [20, 25]

O Modelo de *Markov* representa uma modelagem estocástica da voz. Durante o treinamento, as observações são associadas aos estados do modelo, de acordo com as probabilidades de transições entre os

estados e as probabilidades das observações dado o estado. Essas observações são segmentadas conforme sua semelhança acústica (velocidade, entonação, emoção, etc). Após o treinamento de todas as locuções, as observações são separadas em estados, levando em consideração as variações temporais das seqüências das observações. Devido a existência dessa variabilidade acústica, os vetores de características da voz podem apresentar uma distribuição multimodal. Essa distribuição fica representada no espaço espectral de cada estado, por uma densidade de probabilidade que pode ser: discreta, ou contínua ou ainda semi-contínua.

a) Modelagem Contínua

Em algumas aplicações, as observações nos estados são freqüentemente sinais contínuos (ou vetores). Quando é possível converter tais representações contínuas em uma seqüência de símbolos discretos por meio da quantização vetorial, palavra-código ou outros métodos, poderá ocasionar degradações associadas com tal discretização do sinal contínuo. Consequentemente seria vantajoso utilizar o HMM com densidades de observações contínuas, para modelar diretamente sinais contínuos.

Quais as vantagens do modelamento contínuo? Esse modelamento supõe que cada unidade fonética possa ser modelada independentemente. Uma vez que a segmentação das unidades fonéticas seja fornecida, os parâmetros do HMM são estimados diretamente. Não existe a necessidade de definir um grupo comum de densidades. Usando a suposição de misturas de Gaussianas, todas as densidades de observação podem ser aproximadas em qualquer precisão. Por isto não é feita nenhuma suposição sobre a forma das densidades. Uma das vantagens do modelamento independente das unidades fonéticas é a capacidade de gerar um grau de resolução acústica diferente para cada unidade. [20, 21]

Para utilizar o modelamento contínuo, deve-se supor que o espaço acústico tem uma forma paramétrica, e se esta forma paramétrica for composta por uma mistura de K gaussianas, ela fica definida como

$$b_j(O_i) = \sum_{k=1}^{M} c_{jk} N(o_i, \mu_{jk}, U_{jk}) \qquad (3.7)$$

onde: c_{jk} é um fator de peso para cada gaussiana com média u_{jk} e covariância U_{jk}. O fator de peso, deve satisfazer as condições estocásticas, isto é,

$$c_{jk} \geq 0, \qquad 1 \leq j \leq n \qquad 1 \leq k \leq M \qquad (3.8)$$

Durante o treinamento, para reestimar b é necessário reestimar c, μ e U, usando um conjunto de fórmulas que serão vistas no decorrer deste compendio. O problema dessa aproximação é que os parâmetros não são compartilhados pelos estados. Isso indica que, para que o sistema seja bem treinado, há necessidade de uma grande quantidade de dados.

b) Modelagem Discreta [20]

É a modelagem mais usada no reconhecimento da voz em vocabulários extensos. O modelo discreto é baseado em um livro-código de tamanho finito (ou múltiplos livros-códigos) e um grupo de HMM's discretos. O modelamento é feito em duas fases. Na primeira, o livro-código é elaborado sem o conhecimento do contexto lingüístico da base de dados de treinamento. Na segunda, supõe-se um grupo fixo de livros-códigos nos quais é aplicado o procedimento de reestimação de *Baum-Welch* para estimar os parâmetros do modelamento discreto.

O tamanho do livro-código é comumente escolhido com 256 palavras-códigos. Quando múltiplos livros-códigos são usados para modelar o espaço acústico, cada livro-código terá o tamanho de 256. A aproximação feita pelo modelamento discreto gera erros de quantização e a resolução acústica é limitada. Uma maneira para aumentar a resolução no campo acústico é aumentar o tamanho do livro-código; contudo isso resulta na diminuição do desempenho de reconhecimento.

c) Modelagem Semi-Contínua

Esse modelo tem todas as vantagens do modelamento discreto e ainda ameniza alguns problemas surgidos com a quantização. Contudo, ele ainda tem uma resolução acústica limitada que diminuirá o desempenho do modelo.

3.3.4 Distribuição da Probabilidade Inicial

$$\pi_i = \mathrm{P}\left(q_1 = i\right), \quad 1 \leq \mathrm{i} \leq \mathrm{N} \tag{3.9}$$

onde $\pi = \pi_i$, é a probabilidade do estado i iniciar o processo.

Como pode ser visto, a especificação completa de um *HMM* requer a especificação de dois modelos, *N* e *M*, a especificação dos símbolos das observações, e a especificação dos três grupos de medidas de probabilidade *A*, *B* e π. Por conveniência, utiliza-se a notação compacta

$$\lambda = (A, B, \pi) \tag{3.10}$$

para indicar o grupo de parâmetros de um modelo. Este grupo de parâmetros, é claro, define uma medida de probabilidade para *O*, isto é, $P(O/\lambda)$. A e B são matrizes e obedecem às propriedades probabilísticas. [9]

3.4 SUPOSIÇÕES NECESSÁRIAS PARA A UTILIZAÇÃO DO HMM NO RAV [1, 26]

Para tornar viável a utilização do HMM no reconhecimento, serão feitas as seguintes suposições:

- **Hipótese de *Markov* de Primeira Ordem**: De acordo com a cadeia de *Markov*, o próximo estado do HMM poderá depender dos K estados passados, mas, normalmente esse modelo não é utilizado devido ao aumento no desempenho não compensar ao aumento obtido na complexidade do algoritmo. Desse modo, supõe-se que o próximo estado somente é dependente do estado atual, assim, o modelo resultante torna-se um *HMM* de Primeira Ordem,

$$P(q_t = j/q_{t-1} = i, q_{t-2} = k, ...) = P(q_t = j/q_{t-1} = i) \tag{3.11}$$

onde *i* é o estado atual e *j* é o próximo estado. Desse modo, a transição será independente do tempo, a complexidade torna-se menor e a matriz de transição entre os estados, a_{ij}, será igual ao lado esquerdo da igualdade mostrada na Equação 3.11.

- **Suposição de estacionaridade**: é suposto que as probabilidades de transição entre os estados são independentes do instante em que ocorrem, isto é

$$P(q_{t1+1} = j/q_{t1} = i) = P(q_{t2+1} = j/q_{t2} = i) \tag{3.12}$$

- **Hipótese da independência**: é suposto que as observações, dentro de uma seqüência de observações, são independentes entre si.

$$O=\{o_1, o_2, ..., o_T\} \tag{3.13}$$

$$P(O/q_1, q_2, ..., q_T, \lambda) = \prod_{t=1}^{T} P(O/q, \lambda) \tag{3.14}$$

- **A probabilidade a priori de um modelo pode ser separadamente estimada** - um modelo de linguagem pode ser obtido sem o conhecimento dos dados acústicos.

3.5 OS TRÊS PROBLEMAS BÁSICOS ENCONTRADOS NO HMM[9]

Dada a forma do HMM, os três problemas básicos que devem ser resolvidos para que se possa utilizar o modelo são:

1º Dada a seqüência de observações, $O = (o_1\ o_2\ ...\ o_T)$, e um modelo $\lambda = (A, B, \pi)$, como calcular $P(O / \lambda)$, eficientemente isto é, a probabilidade da seqüência de observação, dado o modelo ?

2º Dada a seqüência de observações $O = (o_1\ o_2\ ...\ o_T)$ e o modelo λ, como escolher a seqüência de estados correspondente, $q = (q_1\ q_2\ ...\ q_T)$, que seja ótima ?

3º Como ajustar o modelo $\lambda = (A, B, \pi)$ para maximizar $P(O \setminus \lambda)$?

Estes três problemas serão resolvidos no desenvolvimento do algoritmo de treinamento do HMM, visto que não existe qualquer algoritmo ou método que os resolva analiticamente.

3.6 DESENVOLVIMENTO DO ALGORITMO DE TREINAMENTO DO HMM [2]

Para o desenvolvimento do algoritmo, deve-se seguir as seguintes fases:

Fase 1: Inicialização

- estrutura do modelo $\begin{cases} \text{parâmetros fixos} \\ \text{parâmetros variáveis} \end{cases}$

Fase 2: Treinamento

- estimação dos parâmetros
- reestimação dos parâmetros

Fase 3: Reconhecimento

3.6.1 Inicialização [2]

Esta fase tem como finalidade apresentar os parâmetros do modelo que serão treinados pelo *HMM*. Estes parâmetros são classificados em:

- **Fixos**: número de estados e de misturas por estado, que não se alteram durante o treinamento.
- **Variáveis**: são a matriz de probabilidade de transição, o vetor média e a matriz covariância dos grupos.

3.6.1.1 Parâmetros Fixos [2]

a) Importância do Número de estados

O objetivo dos modelos acústicos é proporcionar um método eficaz para o cálculo da verossimilhaça. Em princípio, as distribuições de probabilidade podem ser estimadas a partir de várias repetições de cada palavra do dicionário, entretanto, em sistemas com vocabulários grandes isso se torna impraticável. Consequentemente, as seqüências de palavras são decompostas em sons básicos denominados de Unidades Fonéticas (fones, difones, trifones). Na língua inglesa, por exemplo, existem 60 fones básicos, e destes formam-se palavras e seqüência de modelos.

Pode-se representar cada unidade fonética (UF) por um HMM, com três estados e uma topologia do tipo Esquerda-Direita. Estudos indicam que variações nesse modelo produzem muito pouco ou nenhum efeito no desempenho do sistema. De acordo com *Rabiner*, de 2 a 10 estados por fonema são suficientes, ou, pode-se considerar o número médio de observações nas elocuções.

Como pode ser visto, essa característica varia com o pesquisador.

b) Número de Misturas

Na construção de histogramas das observações pertencentes a um determinado estado, verifica-se a presença de uma função de densidade de probabilidade multimodal. Desse modo, o número de misturas em cada estado deve ser maior que a unidade.

3.6.1.2 Parâmetros Variáveis [2]

Devem seguir as seguintes regras:

- Os valores atribuídos a matriz *A* podem ser inicialmente aleatórios.
- O coeficiente de mistura**,** C_{jm}**,** é calculado por meio de uma ponderação estocástica no somatório das gaussianas.
- O cálculo da matriz de probabilidade das observações, b_j, influencia no desempenho do treinamento do modelo; sendo assim não se atribuem valores aleatórios para a média e a covariância.

3.6.2 Treinamento [9]

Deseja-se calcular a probabilidade da seqüência de observações, $O = (o_1, o_2, ..., o_T)$, dado o modelo λ, isto é, P (O / λ).O caminho mais direto é por meio da enumeração de todas as possíveis seqüências de estado T (número de observações), onde q_T é uma seqüência fixa de estados.

$$q_t = (q_1 \; q_2 \ldots \; q_T) \tag{3.15}$$

onde q_1 é o estado inicial. A probabilidade da sequencia de observações O dada sequencia de estados é

$$P(O/q,\lambda) = \prod_{i=1}^{T} P(o_i/q_i,\lambda) \tag{3.16}$$

Como é suposto que as observações sejam independentes entre si, tem-se que:

$$P(O/q,\ \lambda) = b_{q1}(o_1) b_{q2}(o_2)..b_{qT}(o_T) \quad \text{e} \tag{3.17}$$

A probabilidade da seqüência de estados q pode ser escrita como:

$$P(q/\lambda) = \pi_{q1} a_{q1q2} a_{q2q3} ... a_{qT-1} a_{qT} \tag{3.18}$$

A probabilidade conjunta de O e q, é simplesmente o produto dos dois termos acima, isto é,

$$P(O,q/\lambda) = \frac{P(O,q,\lambda)}{P(\lambda)} \tag{3.19}$$

$$P(O,q/\lambda) = \frac{P(O/q,\lambda)P(q/\lambda)P(\lambda)}{P(\lambda)} = P(O/q,\lambda)P(q/\lambda) \tag{3.20}$$

e como:

$$P\left(\frac{O}{\lambda}\right) = \sum_{\text{todos os q}} P(O,q/\lambda) \tag{3.21}$$

Substituindo-se as Equações 3.18 e 3.19 em 3.21 têm-se que,

$$P(O/\lambda) = \sum_{\text{todos q}} \pi_{q1} b_{q1}(o_1) a_{q1q2} b_{q2}(o_2)..a_{qT-1qT} b_{qT}(o_T) \tag{3.22}$$

A interpretação da computação acima é a seguinte: Inicialmente (no tempo t=1) no estado q_1 com probabilidade π_{q1}, o HMM gera o símbolo o_1 (nesse estado) com probabilidade $b_{q1}(o_1)$. O *clock* varia do tempo t para o t+1 e uma transição para o estado q_2 do estado q_1 com probabilidade a_{q1q2} é realizada gerando um símbolo o_2 com probabilidade $b_{q2}(o_2)$. Esse processo continua dessa maneira até a última transição ser realizada.

De acordo com a definição da Equação 3.21, o cálculo de P (O/λ) envolve (2T-1). 2^{NT} multiplicações, e N^T-1 adições. Claramente necessita-se de um procedimento mais eficiente para calcular P(O/λ)

3.7 UM PROCEDIMENTO MAIS EFICIENTE PARA O CÁLCULO DA P (O/ λ)

Como a convergência dos algoritmos apresentados é muito sensível aos valores iniciais de $b_j(o)$, foi utilizado o algoritmo "Segmental K-means", para estimar os valores do modelo λ.

3.7.1 Procedimento [2]

Para uma dada entrada, *O*, *P (O / λ)* poderá ser uma função não linear com muitos máximos locais em um espaço multidimensional. Desse modo para o HMM convergir, o modelo atual, λ^{-}, deverá ser igual a um modelo ótimo, λ^{*}, que corresponda ao máximo global da função. Essa igualdade é obtida por meio de um processo iterativo, na reestimação, em que os parâmetros são atualizados. Se o algoritmo não convergir, deverá ser inicializado várias vezes com diferentes conjuntos de parâmetros iniciais.

3.7.2 Valores Iniciais

Os valores iniciais influenciam diretamente a convergência do treinamento do modelo e são utilizados como uma base probabilística para a primeira segmentação das repetições. Assim, será necessária, a utilização do algoritmo abaixo para a estimação desses valores.

- Para cada repetição, as T observações são divididas de acordo com o número de estados N.
- Agrupa-se as observações nos estados, utilizando o algoritmo *K-means* em cada estado para agrupar as observações, de acordo com o número de misturas, em M grupos.
- Para cada grupo encontra-se a média, a covariância e os coeficientes de mistura.

Depois da segmentação, os valores iniciais são ignorados.

3.7.3 Estimação dos Parâmetros

Para cada repetição:

- Segmenta-se as seqüências de observações nos estados.
- Utiliza-se o algoritmo de *Viterbi* para encontrar a melhor seqüência de estados.
- Agrupam-se os vetores nos estados de acordo com a segmentação de *Viterbi*.

Após a segmentação das observações de cada repetição, pelo algoritmo de *Viterbi*, consegue-se a melhor seqüência de estados.

Os parâmetros $a_{ij}^{\wedge}$ estimados, são obtidos por meio da contagem do número das transições do estado *i* para o estado *j* , dividido pelo número de todas as transições feitas a partir do estado i inclusive.

$$\hat{a} = \frac{\text{número de transições do estado i para o estado j}}{\text{número de transições do estado i (inclusive)}} \tag{3.22}$$

TABELA 3.1: Exemplo de possíveis caminhos ótimos, dado quatro repetições. A distribuição das observações nos estados está relacionada com a matriz de transição e com a matriz de probabilidade das observações.

REP 1	[o_1^1 o_2^1 o_3^1	o_4^2 o_5^2	o_6^3 o_7^3	O_8^4 o_9^4	o_{10}^5	... o_T^S]
REP 2	[o_1^1 o_2^1	o_3^2 o_4^2 o_5^2	o_6^3	o_7^4 o_8^4 o_9^4	o_{10}^5 o_{11}^5	... o_T^S]
REP 3	[o_1^1 o_2^1 o_3^1	o_4^2 o_5^2 o_6^2	o_7^3 o_8^3 o_9^3	o_{10}^4	o_{11}^5	... o_T^S]
REP 4	[o_1^1 o_2^1	o_3^2 o_4^2	o_5^3 o_6^3 o_7^3	o_8^4 o_9^4 o_{10}^4	o_{11}^5 o_{12}^5 o_{13}^5	... o_T^S]

Da Tabela 3.1, pode-se ver que:

$$a_{12} = \frac{4}{10} \qquad\qquad a_{23} = \frac{4}{10}$$

Após o agrupamento dos vetores em M grupos, são obtidos: os parâmetros média, covariância e coeficientes das Gaussianas, para cada estado. A média, μ, é simplesmente estimada pela média de todas as observações pertencentes àquela gaussiana; o mesmo para a covariância, U. O coeficiente de misturas será igual ao número de observações classificado no grupo dividido pelo número total de observações classificado naquele estado.

3.7.4 Reestimação dos Parâmetros [1]

Para a reestimação dos parâmetros foi utilizado o método de *Baum-Welch*. Este método baseai-se nas variáveis *Forward* e *Backward*. Essas variáveis serão utilizadas somente para a reestimação dos parâmetros do modelo.

Com os novos parâmetros reestimados, aplica-se novamente o algoritmo de *Viterbi* e um novo valor de verossimilhança é encontrado, $P(O,q/\lambda^{-})$.

A nova verossimilhança é comparada com a anterior; se a diferença exceder um certo limiar, os parâmetros anteriores serão substituídos pelos atuais, $\lambda = \lambda^{-}$, e todo o treinamento será repetido. Caso contrário, terá convergido e os parâmetros serão admitidos como os do modelo treinado.

$$\left|P(O/\lambda^{-}) - P(O/\lambda)\right| \leq \theta \tag{3.23}$$

3.7.5 Algoritmo de *Viterbi* [1, 9]

O algoritmo de *Viterbi* é utilizado para encontrar a melhor seqüência de estados, $q = (q_1 q_2 ... q_T)$, para uma dada seqüência de observações, $O = (o_1 o_2 ... o_T)$, isto é,

$$\delta_t(i) = \max_{q_1 q_2 ... q_{t-1}} P[q_1 q_2 ... q_{t-1}, q_t = i, o_1 o_2 ... o_t / \lambda] \tag{3.24}$$

onde δ é o maior valor ("maior probabilidade") ao longo de um caminho, no tempo t, ao percorrer as primeiras t observações e terminando no estado i.

Para recuperar a seqüência de estados, torna-se necessária a localização do argumento que maximiza o vetor δ, para cada i e j. Isso é feito através do vetor Ψ(j).

A medida que t aumenta, os valores apresentados pelo algoritmo de *Viterbi* diminuem rapidamente. Para evitar esse *underflow* (o valor se torna tão baixo que ultrapassa a precisão da máquina) utiliza-se o logaritmo da verossimilhança. O procedimento completo para achar a seqüência ótima é mostrado a seguir:

1-Pré-Processamento

$$\tilde{\pi}_i = \log(\pi_i) \qquad 1 \leq i \leq N \tag{3.25}$$

$$\tilde{b}_i(o_t) = \log[b_i(o_t)] \qquad 1 \leq i \leq N,\ 1 \leq t \leq T \tag{3.26}$$

$$a_{ij} = \log(a_{ij}) \qquad 1 \leq i, j \leq N \tag{3.27}$$

2- Inicialização

$$\tilde{\delta}_1(i) = \log(\delta_1(i)) = \tilde{\pi}_i + \tilde{b}_i(o_1), \qquad 1 \leq i \leq N \tag{3.28}$$

$$\Psi_1(i) = 0 \qquad 1 \leq i \leq N \tag{3.29}$$

3- Recursão

$$\tilde{\delta}_t(j) = \log\left(\delta_t(j)\right) = \max_{1 \le i \le N} \left[\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}\right] + \tilde{b}_j(o_t) \tag{3.30}$$

$$\Psi_t(j) = \arg\max_{1 \le i \le N} \left[\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}\right], \qquad 2 \le t \le T,\ 1 \le j \le N \tag{3.31}$$

4- Término

$$\tilde{P}^* = \max_{i \le i \le N} \left[\tilde{\delta}_T(i)\right] \tag{3.32}$$

$$q_T^* = arq \max_{1 \le i \le N} \left[\tilde{\delta}_T(i)\right] \tag{3.33}$$

A medida resultante, calculada para o modelo λ , de uma seqüência de observações será

$$\mathbf{P\,(\,O, Q\,/\,\lambda\,)} \tag{3.34}$$

Para utilizar este algoritmo, deve-se conhecer os seguintes parâmetros:

- $\pi_i \Rightarrow$ probabilidade do estado inicial
- $B = \{\, b_j\,(\,O_t\,)\,\} \Rightarrow$ função densidade de probabilidade
- $A = \{\, a_{ij}\,\} \Rightarrow$ matriz de transição entre os estados
- $N \Rightarrow$ número de estados do HMM

3.7.6 Método de *Baum Welch* (Procedimento *Forward- Backward*) [4]

A reestimação das matrizes *A* e *B* não é um problema trivial, pois, é preciso ajustá-las de modo a maximizar P (O/λ). Com o método *de Baum Welck* consegue-se maximizar localmente a probabilidade da seqüência de observações dado o modelo λ. Inicialmente P (O / λ) é decomposto, em um tempo arbitrário t,

$$P(O/\lambda) = \sum_{j=1}^{N} P(O, q(t) = S_j/\lambda) = \sum_{j=1}^{N} P(o_1 o_2 ... o_T, q(t) = S_j/\lambda) \tag{3.35}$$

posteriormente, supõe-se que as observações são independentes entre si,

$$= \sum_{j=1}^{N} \frac{P(o_{t+1} o_{t+2} ... o_T) P(o_1 o_2 ... o_t, q_t = S_j, \lambda)}{P(o_1 o_2 ... o_t, q_t = S_j, \lambda)} \times \frac{P(o_1 o_2 ... o_t, q_t = S_j, \lambda)}{P(\lambda)} \tag{3.36}$$

$$= \sum_{j=1}^{N} P(o_{t+1}o_{t+2}...o_T / o_1o_2...o_t, q_t = S_j, \lambda).P(o_1o_2...o_t, q_t = S_j / \lambda) = \quad (3.37)$$

Como a verossimilhança condicional é independente de $o_1o_2...o_t$, têm-se que

$$= \sum_{j=1}^{N} P(o_{t+1}o_{t+2}...o_T / q_t = S_j, \lambda).P(o_1o_2...o_t, q_t = S_j / \lambda) \quad (3.38)$$

De acordo com a equação 3.38, pode-se definir as variáveis *Forward* e *Backward* como:

$$\alpha_t(j) = P(o_1o_2...o_t, q_t = S_j / \lambda) \quad (3.39)$$

$$\beta_t(j) = P(o_{t+1}o_{t+2}....o_T / q_T = S_i, \lambda) \quad (3.40)$$

e conclui-se que:

$$P(O/\lambda) = \sum_{j=1}^{N} \alpha_t(j)\beta_t(j) \quad (3.41)$$

Conhecendo as variáveis *Forward* e *Backward*, pode-se descrever o procedimento de rees-timação dos parâmetros do HMM.

3.7.6.1 Procedimento *Forward* [1]

A variável *Forward*, $\alpha_t(i)$, é definida como

$$\alpha_t(i) = P(o_1o_2...o_t, q_t = i / \lambda) \quad (3.42)$$

isto é, a probabilidade da seqüência parcial de observações, $o_1 o_2...o_t$, (até o tempo t) e o estado i no tempo t, dado o modelo λ. Pode-se resolver $\alpha_t(i)$ indutivamente.

1. Inicialização

$$\alpha_1(i) = \pi_i b_i(o_1), \qquad 1 \le i \le N \quad (3.43)$$

2. Indução

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i)a_{ij}\right] b_j(o_{t+1}), \qquad 1 \le t \le T-1 \quad 1 \le j \le N \quad (3.44)$$

3. Término

$$P(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i) \quad (3.45)$$

3.7.6.2 Procedimento *Backward*

A variável *Backward* , $\beta_t(i)$, é definida como

$$\beta_t(i) = P(o_{t+1}o_{t+2}...o_T / q_T = i, \lambda) \tag{3.46}$$

isto é, a probabilidade da seqüência parcial de observações do tempo t+1 até o final; dado o estado i no tempo t e o modelo λ. Novamente pode-se resolver $\beta_t(i)$ de forma indutiva, como se segue,

1. Inicialização

$$\beta_t(i) = 1, \qquad 1 \le i \le N \tag{3.47}$$

2. Indução

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \qquad t = T-1, T-2, \ldots 1 \tag{3.48}$$

3.7.6.3 Variáveis Utilizadas na Reestimação [1, 2, 9]

Após o cálculo de α e β, defini-se $\xi_\tau(i, j)$ como sendo a probabilidade de estar no estado i no tempo t e no estado j no tempo t +1, dado o modelo e a seqüência de observações, isto é

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j / O, \lambda) \tag{3.49}$$

ou

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \tag{3.50}$$

Defini-se agora $\gamma_t(i, k)$, como sendo a probabilidade de estar no estado i e no grupo K no tem t, dado a seqüência de observação e o modelo.

$$\gamma_t(i, k) = P(q_t = i, m = k / O, \lambda) \tag{3.51}$$

ou

$$\gamma_t(j, k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \right] \bullet \left[\frac{c_{jk} N(o_t, \mu_{jk}, U_{jk})}{\sum_{m=1}^{M} c_{jm} N(o_t, \mu_m, U_{jm})} \right] \tag{3.52}$$

A Figura 3.2 ilustra todas as possíveis seqüências de estados, incluindo o estado S_i, observando uma determinada gaussiana. $\gamma_t(i,k)$ é o cálculo do valor esperado do número de vezes que uma específica observação ocorre em uma determinada Gaussiana k e no estado S_i

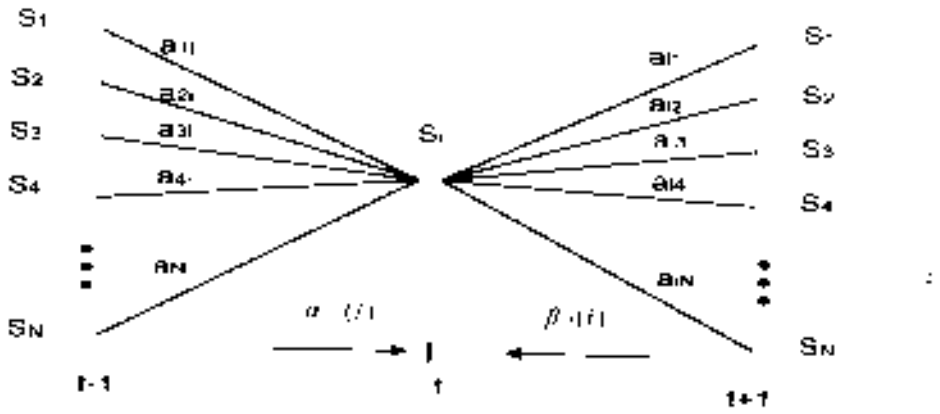


FIGURA 3.2: Seqüência de operações requerida para a computação da variável $\gamma_t(i,k)$.

Ao somar todas as transições feitas, do estado i para o estado j, em $\gamma_t(i,k)$ têm-se γ_t (i), que pode ser definido como: a probabilidade de estar no estado i no tempo t, dada a seqüência de observações e o modelo; consequentemente, pode-se relacionar γ_t (i) com $\xi_t(i,j)$ pelo somatório em todos os j's, isto é,

$$\gamma_t = \sum_{j=1}^{N} \xi_t(i,j) \tag{3.53}$$

O somatório da Equação 3.53, origina uma quantidade que pode ser interpretada como o número de vezes que o estado i é visitado. Se esse mesmo somatório variar de t=1 até t= T - 1, essa nova quantidade pode ser interpretada como o número de transições a partir do estado S_i. Similarmente, pode-se dizer que: um somatório em $\xi_t(i,j)$ no índice t (de t=1 até t = T -1), pode ser interpretado como o número de esperado de transições do estado s_i para o estado s_j.

$$\sum_{t=1}^{T-1} \gamma_t(j) = \text{número esperado de transições a partir do estado i em O} \tag{3.54}$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{número esperado de transições do estado i para o estado j em O} \tag{3.55}$$

3.7.7 **Parâmetros do Modelo** [1, 5, 9]

Com base nas equações até aqui descritas, consegui-se elaborar um método capaz de reestimar os parâmetros do HMM.

$$\bar{\pi}_i = \text{número de vezes que ocorre o etado i no tempo t} = 1 \qquad \gamma_1 = (i) \tag{3.56}$$

$$\bar{a}_{ij} = \frac{\text{número esperado de transições do estado i para o estado j}}{\text{número esperado de transições a partir do estado i}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{3.57}$$

$\bar{b}_j(O) =$ probabilidade da observação dados o estado j

$$\bar{b}_j(O) = \sum_{k=1}^{M} c_{jk} N(O, u_{jm}, U_{jm}) \qquad 1 \le j \le N \tag{3.58}$$

onde pode ser provado [1], que seus parâmetros são iguais a

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{3.59}$$

$$\mu_{jk} = \frac{\text{número esperado de ocorrer a Gaussiana K no estado j ponderada pela observação o}}{\text{número esperado de estar no estado j e na mistura k}} \tag{3.60}$$

$$\mu_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) o_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{3.61}$$

$$U_{jk} = \frac{\text{número esperado a Gaussiana k no estado j ponderado pela covariância}}{\text{número esperado de estar no estado j na Gaussiana k}} \tag{3.62}$$

$$U_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \left[(o_t - \mu_j)(o_t - \mu_j)^T \right]}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{3.63}$$

3.7.7.1 Normalização [22, 23]

Os valores dos parâmetros do HMM encontrados anteriormente, para valores de **t** suficientemente grandes, podem exceder a precisão da máquina. A solução é utilizar uma ponderação que seje dependente de t, para que o cálculo fique na faixa dinâmica do computador. No final da computação essa ponderação será anulada. Primeiramente encontra-se o coeficiente de normalização para as variáveis $\alpha_t(i)$

$$c_t = \frac{1}{\sum_{i=1}^{N} \alpha_t \; i} \tag{3.64}$$

normalizando a variável:

$$\tilde{\alpha}(i) = \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_t(i)} \tag{3.65}$$

e a Equação 3.44 tornar-se

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \tilde{\alpha}_t(i) a_{ij} \right] b_j(o_{t+1}) \tag{3.66}$$

O coeficiente de escala para as variáveis $\beta_t(i)$ é

$$c_t = \frac{1}{\sum_{j=1}^{N} \beta_t(i)} \tag{3.67}$$

normalizando a variável:

$$\tilde{\beta}_t(i) = \frac{\beta_t(i)}{\sum_{j=1}^{N} \beta_t(i)} \tag{3.68}$$

a Equação 3.48 tornar-se

$$\beta_{t-1}(j) = \sum_{i=1}^{N} a_{ij} b_i(o_t) \tilde{\beta}_t(i) \quad , \quad 1 \le j \le N \tag{3.69}$$

Na fórmula de a_{ij} , os termos do fator de escalonamento se cancelam fazendo

$$\tilde{\alpha}_t(i) = c_t \alpha_t(i) \tag{3.70}$$

$$\tilde{\beta}_{t+1}(i) = c_{t+1} \beta_{t+1}(i) \tag{3.71}$$

$$c_t \, . \, c_{t+1} = C_T \tag{3.72}$$

d) Número de seqüências para treinamento

Até esse ponto, o cálculo de a_{ij} refere-se a uma seqüência de observações. Porém, poderá existir uma baixa probabilidade da ocorrência de uma observação em um determinado estado impossibilitando o algoritmo de obter boas estimativas dos parâmetros do modelo. Dessa forma, para uma melhor obtenção são utilizadas seqüências múltiplas de observações, isto é,

$$O = [\, O^{(1)}, O^{(2)}, \ldots O^{(R)}] \tag{3.73}$$

Sendo R o número de repetições da locução usada para o treinamento. Supõe-se que as repetições são independentes entre si:

$$P(O/\lambda) = \prod_{o=1}^{R} P\left(O^{(o)}/\lambda\right) \tag{3.74}$$

Com $\alpha^o{}_t(j), \beta^o{}_t(j)$ representando os parâmetros da o-ésima locução,

d-1) a probabilidade de transição finalmente se torna:

$$\bar{a}_{ij} = \frac{\sum_{o=1}^{R} \sum_{t=1}^{T_0-1} \tilde{\alpha}_t(i) a_{ij} b_j\left(o^o_{t+1}\right) \tilde{\beta}^o{}_{t+1}(j)}{\sum_{o=1}^{R} \sum_{t=1}^{T-1} \sum_{j=1}^{N} \tilde{\alpha}^o_t(i) a_{ij} b_j\left(o^o_{t+1}\right) \tilde{\beta}^o_{t+1}(j)} \tag{3.75}$$

Para a obtenção das fórmulas para a reestimação, devem ser feitas algumas simplificações.

Fórmula simplificada da Matriz de Transição

$$\bar{a}_{ij} = \frac{\sum_{o=1}^{R} \sum_{t=1}^{T_o-1} \tilde{\alpha}^o_t(i) a_{ij} b_j\left(o^o_{t+1}\right) \tilde{\beta}^o_{t+1}(j)}{\sum_{o=1}^{R} \sum_{t=1}^{T} \tilde{\alpha}^o_t(i) \tilde{\beta}^o_t(j) / c_t} \tag{3.76}$$

d-2) Coeficiente Reestimado

- forma expandida

$$\bar{c}_{jk} = \frac{\sum_{o=1}^{R}\sum_{t=1}^{T-1}\sum_{i=1}^{N}\tilde{\alpha}_t^{o}(j)a_{ji}b_i\left(o_{t+1}^{o}\right)\tilde{\beta}_{t+1}^{o}(j)D^{o}{}_{jk}}{\sum_{o=1}^{R}\sum_{t=1}^{T-1}\sum_{i=1}^{N}\tilde{\alpha}_t^{o}(j)a_{ji}b_i\left(o_{t+1}^{o}\right)\tilde{\beta}_{t+1}^{o}(j)} \tag{3.77}$$

- forma simplificada

$$\bar{c}_{jk} = \frac{\sum_{o=1}^{R}\sum_{t=1}^{T}\tilde{\alpha}_t^{o}(j)\tilde{\beta}_t^{o}(j)/c_t D^{o}{}_{jk}}{\sum_{o=1}^{R}\sum_{t=1}^{T}\tilde{\alpha}_t^{o}(j)\tilde{\beta}_t^{o}(j)/c_t} \tag{3.78}$$

$$D^{o}{}_{jk} = \left[\frac{c_{jk}N\left(o^{o}{}_t,\mu_{jk},U_{jk}\right)}{\sum_{k=1}^{M}c_{jk}N\left(o^{o}{}_t,\mu_{jk},U_{jk}\right)}\right] \tag{3.79}$$

d-3) Média Reestimada

- Forma Expandida

$$\bar{\mu}_{jk} = \frac{\sum_{o=1}^{R}\sum_{t=1}^{T-1}\sum_{i=1}^{N}\tilde{\alpha}_t^{o}(j)a_{ji}b_i\left(o_{t+1}^{o}\right)\tilde{\beta}_{t+1}^{o}(j)D^{o}{}_{jk}o^{o}{}_t}{\sum_{o=1}^{R}\sum_{t=1}^{T-1}\sum_{i=1}^{N}\tilde{\alpha}_t^{o}(j)a_{ji}b_i\left(o_{t+1}^{o}\right)\tilde{\beta}_{t+1}^{o}(j)D^{o}{}_{jk}} \tag{3.80}$$

- Forma Simplificada

$$\bar{\mu}_{jk} = \frac{\sum_{o=1}^{R}\sum_{t=1}^{T}\tilde{\alpha}_t^{o}(j)\tilde{\beta}_t^{o}(j)/c_t D^{o}{}_{jk}o^{o}{}_t}{\sum_{o=1}^{R}\sum_{t=1}^{T}\tilde{\alpha}_t^{o}(j)\tilde{\beta}_t^{o}(j)/c_t D^{o}{}_{jk}} \tag{3.81}$$

d-4) Covariância Reestimada:

- Forma Expandida

$$\bar{U}_{jk} = \frac{\sum_{o=1}^{R}\sum_{t=1}^{T-1}\sum_{i=1}^{N}\tilde{\alpha}_t^{o}(j)a_{ji}b_i\left(o_{t+1}^{o}\right)\tilde{\beta}_{t+1}^{o}(j)D^{o}{}_{jk}\left(o_t^{o}-\mu_{jk}\right)\left(o_t^{o}-\mu_{jk}\right)^{T}}{\sum_{o=1}^{R}\sum_{t=1}^{T-1}\sum_{i=1}^{N}\tilde{\alpha}_t^{o}(j)a_{ji}b_i\left(o_{t+1}^{o}\right)\tilde{\beta}_{t+1}^{o}(j)D^{o}{}_{jk}} \tag{3.82}$$

- Forma Simplificada

$$\bar{U}_{jk} = \frac{\sum_{o=1}^{R}\sum_{t=1}^{T} \tilde{\alpha}_t^o(j)\tilde{\beta}_t^o(j)/c_t D^o{}_{jk}\,(o_t^o - \mu_{jk})(o_t^o - \mu_{jk})^T}{\sum_{o=1}^{R}\sum_{t=1}^{T} \tilde{\alpha}_t^o(j)\tilde{\beta}_t^o(j)/c_t D^o{}_{jk}} \qquad (3.83)$$

Considerando as observações independentes entre si, tem-se

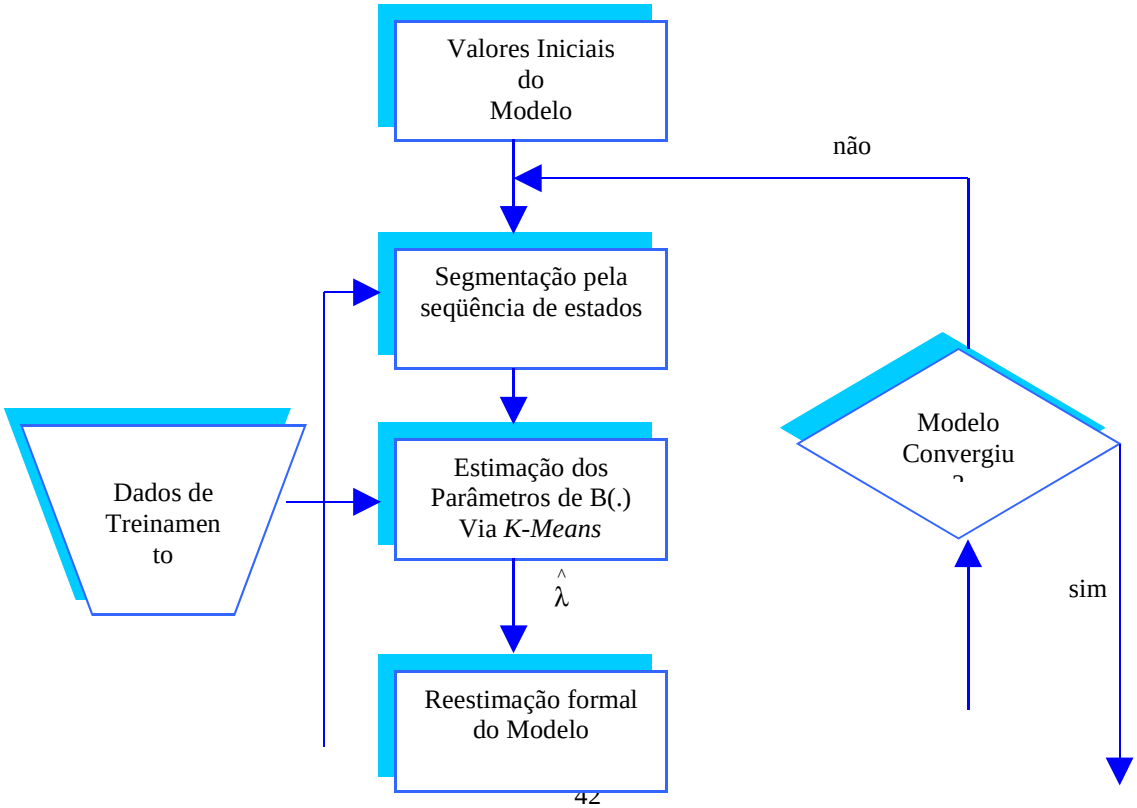
$$B_j(O_t) = \text{Prob}(O_t / \text{estado j}) = \sum_{m=1}^{M} c_{jm} \frac{\prod_{d=1}^{D} e^{-\frac{(O_t(d)-\mu_{jm}(d))^2}{2U_{jm}(d)}}}{(2\pi)^{D/2}\left[\prod_{d=1}^{D} U_{jm}(d)\right]^{1/2}} \qquad (3.84)$$

Assim sendo, obtém-se desta maneira os parâmetros do modelo reestimado.

$$\bar{\lambda} = \left(\bar{A}, \bar{B}\left(\bar{c}_{jm}, \bar{\mu}_{jm}, \bar{U}_{jm}\right), \bar{\pi}_i\right) \qquad (3.85)$$

3.8 TREINAMENTO DE PALAVRAS ISOLADAS UTILIZANDO O HMM [1, 2]

Para facilitar o entendimento do treinamento do HMM, é mostrado abaixo o diagrama em blocos do *Segmental K-means*.



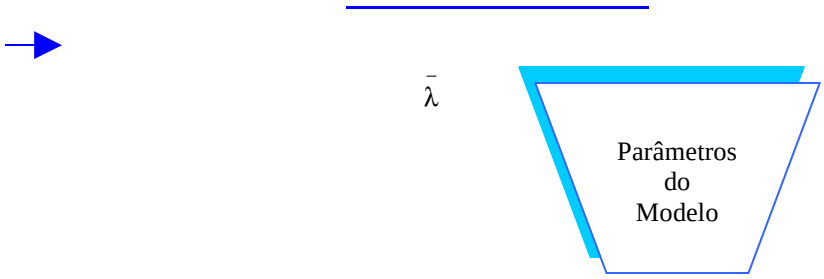


FIGURA 3.3: Algoritmo *Segmental K-means*

3.9 RECONHECIMENTO [2]

Supõe-se que exista um vocabulário de V palavras para ser reconhecido; e que cada palavra foi modelada por um HMM distinto e treinada por um conjunto de K elocuções. Cada elocução (elocução esta, diferente daquelas utilizadas na fase de treinamento) é representada por um seqüência de observações.

Para cada palavra são estimados os parâmetros do modelo que otimizam a verossimilhança. Finalmente, seleciona-se a palavra cuja verossimilhança do modelo é a mais alta. O cálculo dessa verossimilhança é feito pelo algoritmo de *Viterbi*.

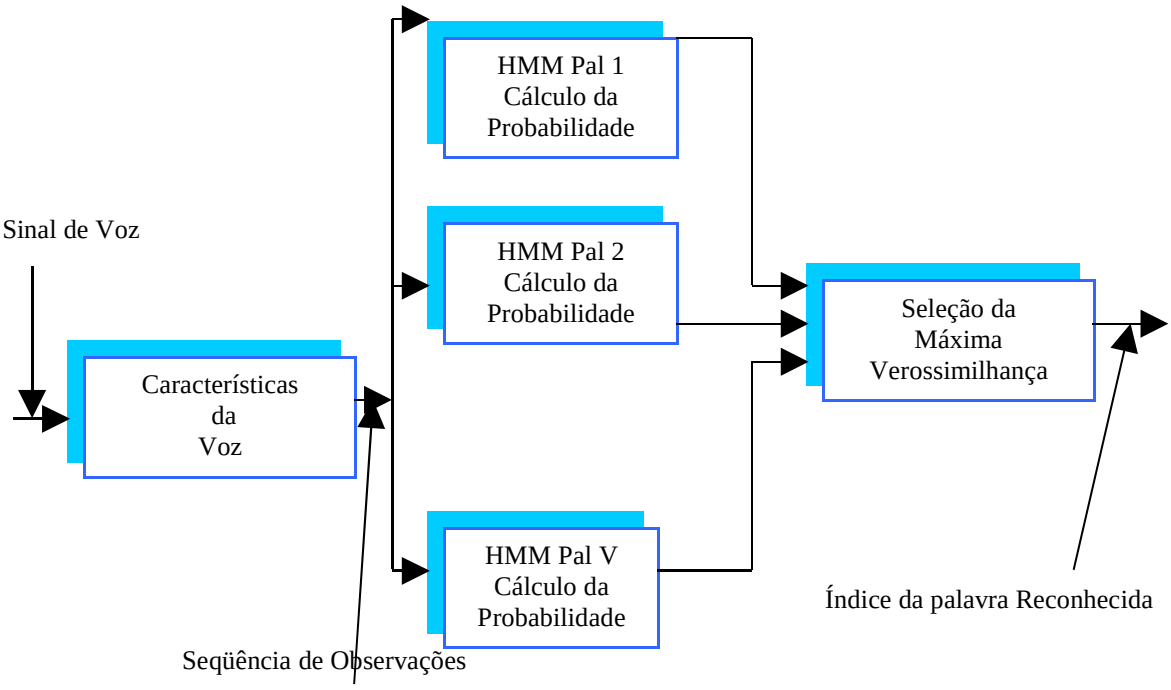


FIGURA 3.4: Diagrama em blocos de um reconhecedor de palavras isoladas com HMM

3.10 INCLUSÃO DA DENSIDADE DE DURAÇÃO DE ESTADO [1, 9, 20]

A densidade de probabilidade da duração de estado, p_i (d), associada com o estado i, com coeficiente de transição, a_{ii} , tem a forma

$$p_i(d) = (a_{ii})^{d-1}(1-a_{ii}) \qquad 0 \le d \le D \tag{3.86}$$

= probabilidade de d consecutivas observaçõs no estado i

Prova-se que essa duração de estado não é apropriada para sinais de voz, uma solução seria o uso de semi-modelos de *Markov* utilizando um algoritmo de *Forward-Backward* modificado. A importância da incorporação dessa densidade é refletida no reconhecimento, pois, em alguns problemas, a qualidade do modelamento é aumentada significantemente quando utiliza-se as densidades de duração de estado. Contudo, esta implementação requer um aumento de D vezes a armazenagem de dados e $D^2/2$ vezes o esforço computacional. Para um D da ordem de 25 (ordem em muitos problemas em problemas de processamento da voz), o esforço computacional é aumentado por um fator de 300. Outro problema com esse modelos é o grande número de parâmetros (D), associados com cada estado, que devem ser estimados, em adição aos usuais parâmetros do HMM (no problema de reestimação, é mais difícil reestimar as variáveis de duração do que as padrões do HMM).

Para diminuir alguns desses problemas, utiliza-se densidades de duração de estado paramétricas em vez das não-paramétricas. Em particular, pode ser incluída a família das Gaussianas de modo que

$$p_i(d) = N(d, \mu_i, \sigma_i^2) \tag{3.87}$$

ou a distribuição Gama com

$$p_i(d) = \frac{\eta_i^{\nu_i} d^{\nu_i - 1} e^{-n_i d}}{\Gamma(\nu_i)} \tag{3.88}$$

com parâmetros ν_i e η_i e com média $\nu_i n_i^{-1}$ e variância $\nu_i n_i^{-2}$.

3.11 PRINCIPAIS LIMITAÇÕES ENCONTRADAS NO HMM PARA APLICAÇÕES EM VOZ [1, 9, 27]

- **A Hipótese de *Markov* de Primeira Ordem** - todas as probabilidades dependem somente do estado corrente. Uma conseqüência é que os HMM's têm dificuldade em modelar coarticulações .
- **Independência entre as Observações**: com essa hipótese o HMM examina uma observação por vez desprezando há correlação entre observações adjacentes, perdendo completamente a informação do

contexto. Uma maneira de amenizar esse problema é, a utilização de características que trazem informações dessa correlação (*Delta Cepstrum*, etc.)

- Os modelos de densidade de probabilidade dos HMM's (contínuo, discreto e semi-contínuo) são sub-ótimos, como já foi visto. [1, 9]

CAPÍTULO IV

REDES NEURAIS

Esse capítulo tem como objetivo descrever os conceitos básicos da Rede Neural. Dentre os assuntos de fundamental importância para o conhecimento de uma RNA são abordados os seguintes tópicos: redes MLP, tipos de treinamento e o algoritmo *Backpropagation.*

4.1 CONCEITOS BÁSICOS DA REDE NEURAL ARTIFICIAL

A célula nervosa elementar, o neurônio, é a unidade fundamental do sistema neural biológico.[6] O neurônio é composto, basicamente, de três regiões: o corpo celular, o axônio e os dendritos. Os dendritos tem por função receber as informações oriundas de outros nodos, e conduzi-las até o corpo celular. No corpo celular a informação é processada e novos impulsos são gerados. Estes impulsos são transmitidos a outros nodos, passando através do axônio até os dendritos dos nodos seguintes. O ponto de contato entre a terminação axônica de um neurônio e o dendrito de outro é chamado sinapse. É pela sinapse que os nodos se unem funcionalmente, formando as redes neurais.[15]

As redes neurais artificiais foram concebidas de forma a emular em um computador, a estrutura e a funcionalidade do cérebro, dessa forma, os neurônios passam a ser representados como simples elementos de processamento, os dendritos como interconecções, terminais de entrada, as sinapses como pesos e o axônio pelos terminais de saída. O processo de combinação do sinais e geração de uma saída para o neurônio são modelados por uma função de transferência, as sinapses de cada conecção são representadas por pesos que variam durante o treinamento. [15]

Esse modelo oferece as seguintes propriedades que serão úteis durante o RAV:[1, 28]

- **Não-Linearidade** - As redes podem operar funções não lineares e não paramétricas de suas entradas, habilitando assim em desenvolver funções complexas de transformação de dados**.**
- **Informação Contextual** - O conhecimento é representado por várias estruturas e estados de ativação de uma rede neural. Todos os neurônios são potencialmente afetados pela atividade global de todos os outros neurônio da rede. Conseqüentemente, a informação contextual é desenvolvida naturalmente pela rede neural.
- **Adaptabilidade** - A rede neural possui a capacidade de adaptar seus pesos de acordo com as variações do ambiente em que se encontra. Em particular, uma rede neural treinada para operar em um específico ambiente pode ser facilmente retreinada, com poucas modificações, para operar em condições ambientais diferentes.
- **Robustez** - As redes são tolerantes a falhas e dados ruidosos.
- **Generalização** - As redes não apenas memorizam os dados treinados, mas também podem generalizar para novos padrões. Isso é essencial no reconhecimento da voz, porque os padrões acústicos nunca são exatamente os mesmos.
- **Paralelismo** - As redes neurais são altamente paralelas por natureza, dessa forma são ideais para implementação em computadores de processamento paralelo, permitindo um rápido processamento.

4.2 MODELO NEURONAL

Um neurônio é uma unidade de processamento de informação que é fundamental para a operação de uma rede neural. A Figura 4.1 mostra o modelo para um neurônio. Pode-se identificar três elementos básicos do modelo:

- um grupo de sinapses, cada qual caracterizada por um peso. Por exemplo, o sinal x_j, na entrada da sinapse ***j*,** conectado ao neurônio *k* é multiplicado por um peso w_{kj}, onde *k* refere-se ao neurônio em questão e o ***j*** à sinapse pela qual o peso refere-se;
- uma função de ativação que processa os estímulos ponderados pelos respectivos pesos e mede o estado de ativação para o neurônio;
- uma função de propagação que se encarrega de propagar o estado de ativação do neurônio para os outros que estão conectados ao mesmo.

Em termos matemáticos, pode-se descrever um neurônio k da seguinte maneira:

$$v_k = \sum_{j=1}^{p} w_{kj} x_j \tag{4.1}$$

$$y_k = \varphi(v_k - \theta_k) \tag{4.2}$$

onde x_1, x_2, ..., x_p são os sinais entrantes que representam os dendritos ; w_{k1}, w_{k2}, ... ,w_{kp} são os pesos sinápticos do neurônio k que representam as sinapses; v_k é o estado de ativação do j-ésimo neurônio; θ_k é o limiar; φ (.) é a função de ativação; e y_k é a sinal sainte do neurônio k.

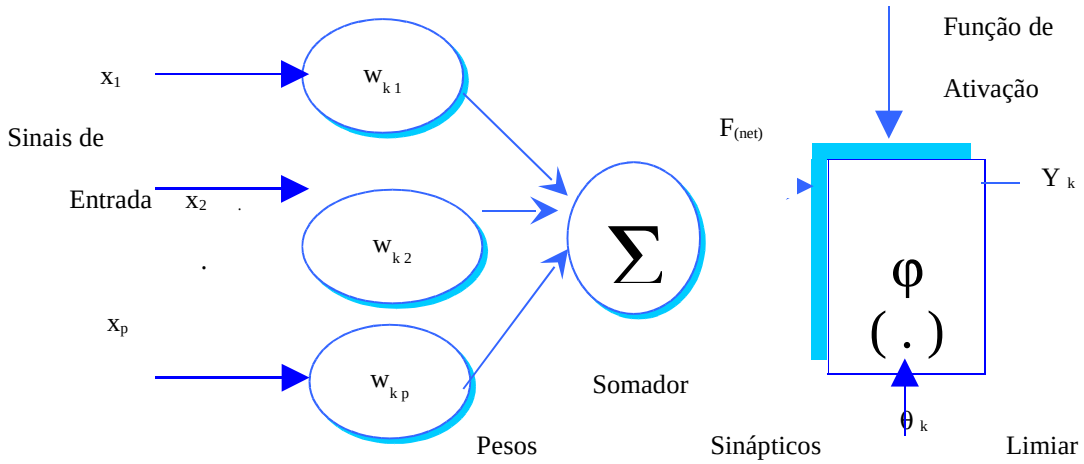


FIGURA 4.1: Modelo não-linear de um neurônio

4.3 CONCEITOS BÁSICOS DA REDES MLP

As redes multi-camadas (MLP) surgiram com a finalidade de solucionar o problema das classes que não são linearmente separáveis. O objetivo das MLP's foi alcançado com a utilização das camadas escondidas.[16] Entretanto a solução deste problema gerou outros dois: o primeiro encontra-se no treinamento dos nodos da camada intermediária e o segundo no tempo que a rede leva para convergir em uma solução que muitas das vezes não é a ótima, isto é, o mínimo global não é alcançado. O primeiro problema tem solução como será mostrado no decorrer deste item, entretanto o segundo problema é parcialmente resolvido com a utilização de variações na estrutura e no algoritmo de treinamento da rede.[17]

Para treinar as redes com mais de uma camada foi proposto o método que se baseia no gradiente descendente, desse modo para que este método possa ser utilizado, a função de propagação precisa ser contínua e diferenciável.

4.3.1 Funções de Ativação [6]

A função de ativação define a saída de um neurônio em termos do nível de ativação da sua entrada. Pode-se identificar quatro tipos básicos de funções de ativação:

- **Linear**

$$a_i(x,w) = w^T x = \sum_{j=1}^{n} w_{ij} x_j \tag{4.3}$$

- **Esférica**

$$a_i(x,w) = p^{-2} \sum_{j=1}^{n} \left(x_j - w_{ij}\right)^2 \tag{4.4}$$

- **Mahalanobis**

$$a_i(x,w) = (x - w_i)^T \Omega^{-1} (x - w_i) \tag{4.5}$$

$$\Omega = XX^T \tag{4.6}$$

- **Polinomial**

$$a_i(x,w) = \sum_{j=1}^{n} x_j^{w_{ij}} \tag{4.7}$$

4.3.2 Funções de Propagação [6]

Após os estímulos da entrada terem sido processados pela função de ativação, o estado de ativação é passado para a função de propagação que produz o valor de saída do neurônio. A função matemática de mapeamento do neurônio, chamada função de transferência ,"f(.)", é constituída pela composição das funções de propagação "p(.)" e de ativação "a(.)", ou seja:

$$f(.) = p(.) \times a(.) \tag{4.8}$$

Deve ser observado, que em uma rede com mais de uma camada cujos nodos utilizam funções de propagação lineares é equivalente a uma rede de uma só camada. Desse modo, esta rede somente será capaz de separar classes linearmente separáveis.

As funções de propagação mais utilizadas são:

- **Degrau**

$$y = \begin{cases} 1, x \geq b \\ 0, x < b \end{cases} \tag{4.9}$$

- **Degrau Simétrico**

$$y = \begin{cases} 1, x \geq b \\ -1, x \leq b \end{cases} \quad (4.10)$$

- **Linear**

$$y = x + b \quad (4.11)$$

- **Logística Sigmoidal**

$$y = \frac{1}{1 + e^{-(n+b)}} \quad (4.12)$$

- **Tangente Sigmoidal**

$$Y = \frac{e^{(x+b)} - e^{-(x+b)}}{e^{(x+b)} + e^{-(x+b)}} \quad (4.13)$$

4.3.3 Arquiteturas da Rede MLP [7, 6]

A maneira pela qual os neurônios de uma rede neural são estruturados esta intimamente ligada ao algoritmo de aprendizado utilizado para treinar a rede. Fazem parte da definição da arquitetura os seguintes parâmetros: número de camadas da rede, número de nodos, tipo de conexão entre os nodos e a topologia da rede.

Quanto ao número de camadas da rede, têm-se:

- redes de camada única – só existe os nodos fontes da camada de entrada e qualquer saída da rede. Deve ser observado, que a camada de entrada não deve ser considerada, pois nenhuma computação nela é realizada.
- redes de múltipla camada – existe mais de um neurônio entre alguma entrada e alguma saída da rede.

Quanto ao tipo de conexão entre os nodos:

- *feedforward*, ou acíclica – a saída de um neurônio na i-ésima camada da rede não pode ser usada como entrada de nodos em camada de índice menor ou igual a i;
- *feedback*, cíclica ou recorrente – a saída de um neurônio na i-ésima camada pode ser usada como entrada de nodos em camada de índice menor ou igual a i.

Quanto a sua conectividade:

- rede fracamente conectada – na ausência de algum enlace, entre os neurônio da camada anterior e os da posterior, a rede é definida como parcialmente conectada;
- rede completamente conectada - todos os neurônios, em cada camada da rede, estão conectados a todos os neurônios da camada posterior.

4.3.3.1 Redes Recorrentes [16]

São redes onde o neurônio podem ser direta ou indiretamente realimentados pela sua saída. Cada camada pode conter conexões entre os elementos de processamento da mesma camada, das camadas anteriores e posteriores. Na estrutura recorrente não existe um sentido único para o fluxo dos sinais entre neurônios ou entre camadas.

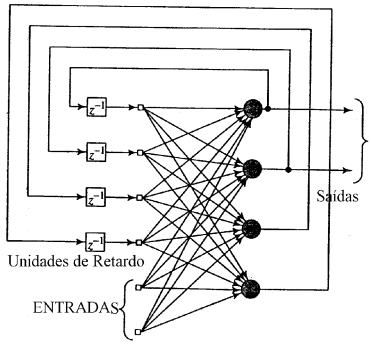


FIGURA 4.2: Rede Recorrente com neurônios escondidos.

4.3.3.2 Redes *Feedforward*

As redes *Feedforward* possuem uma ou mais camadas. As redes com mais de uma camada são caracterizadas pela presença de uma ou mais camadas escondidas. A função dos neurônios escondidos é intervir entre a entrada externa e a saída da rede. Adicionando-se uma ou mais camadas escondidas, a rede fica habilitada à extrair funções estatísticas de altíssimas ordens.[7]

A propagação do sinal da entrada até a saída da rede pode ser descrita da seguinte forma: os nodos fonte da camada de entrada da rede recebem os elementos do modelo, que constituem o sinal entrante, transmitindo-os aos neurônios da segunda camada, isto é, a primeira camada escondida. Os sinais de saída da segunda camada escondida são utilizados como entrada para a terceira camada e assim por diante até a saída final da rede. Geralmente, os neurônios em cada camada da rede tem como suas entradas os sinais de saída dos

neurônios da camada anterior.[17] Na Figura 4.3 tem-se uma rede neural *Feedforward* com uma simples camada escondida do tipo 2-4-10, isto é, 10 nodos fonte, 4 neurônios escondidos, e dois neurônios de saída.

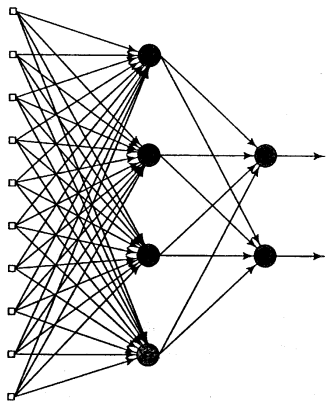


FIGURA 4.3: Rede *Feedforwad* completamente conectada com uma camada escondida e uma camada de saída.

4.3.4 Funcionalidade das Camas em uma Rede MLP [15]

Em uma rede multi-camada, o processamento realizado por cada nodo é definido pela combinação dos processamentos realizados pelos nodos da camada anterior. Quando se segue da primeira camada intermediária em direção à camada de saída, as funções implementadas tornam-se mais complexas. Estas funções definem como será realizada a divisão do espaço. Para uma rede com duas camadas intermediárias, pode-se dizer, a grosso modo, que o processamento em cada uma das camadas dá-se conforme é mostrado abaixo:

- Primeira camada intermediária: cada nodo traça retas no espaço de padrões de treinamento.
- Segunda camada intermediária: cada nodo combina as retas traçadas pelos neurônios da camada anterior conectados a ele, formando regiões convexas, onde o número de lados é definido pelo número de unidades a ele conectados.
- Camada de Saída: cada nodo forma regiões que são combinações das regiões convexas definidas pelos nodos a ele conectados da camada anterior.

As camadas intermediárias de uma MLP funcionam como detetores de características, gerando uma codificação interna dos padrões de entrada, que é utilizada para a definição da saída da rede. Dado o número de camadas intermediárias pode-se dizer que:

- Uma camada intermediária é suficiente para aproximar qualquer função contínua.
- Duas camadas intermediárias são suficientes para aproximar qualquer função matemática.

Deve ser observado que permitir a implementação da função não implica na garantia da implementação da função, isto é, dependendo da distribuição dos dados a rede pode convergir para um mínimo local, demorar demais para encontrar a solução desejada ou não encontrá-la. A utilização de duas ou mais camadas escondidas pode facilitar o treinamento da rede, entretanto esta técnica não é recomendada, pois, a cada vez que o erro medido durante o treinamento é propagado para a camada anterior, ele se torna menos preciso. A única camada que tem uma noção precisa do erro cometido pela rede é a camada de saída. A última camada intermediária recebe apenas uma estimativa sobre o erro. A penúltima camada uma estimativa da estimativa, e assim por diante. Além disso, a utilização de várias unidades intermediárias poderá fazer com que a RNA perca o poder de generalização tornando-a especialista nos dados que lhe foram treinados, por outro lado, um número pequeno de camadas poderá forçar a rede a gastar tempo em excesso tentando encontrar uma representação ótima.

O número de nodos na camada intermediária depende de vários fatores, como:

- número de exemplos de treinamento;
- quantidade de ruído presente nos exemplos;
- complexidade da função obtida;
- distribuição estatística dos dados de treinamento.

Deve-se ter cuidado para não utilizar unidades intermediárias demais, o que pode levar a rede memorizar os padrões de treinamento, ao invés de extrair as características gerais que permitirão a generalização ou reconhecimento de padrões não vistos durante o treinamento, nem um número pequeno, que pode forçar a rede a gastar tempo em excesso tentando encontrar uma representação ótima.

4.3.5 Aprendizado [18]

Constitui-se no ajuste do conjunto de pesos de modo à executar uma determinada tarefa e, basicamente, acontece de duas formas:

- **Supervisionado** - O principal integrante desse tipo de aprendizado é o "professor externo". Pode-se pensar que: um professor tem o conhecimento do ambiente, que é representado por um grupo de entradas e saídas. Esse ambiente, contudo, é desconhecido para a rede neural de interesse. Suponha agora, que o professor e a rede são ambos expostos a um vetor de treinamento, isto é, um exemplo. O professor informa a rede, a saída desejada para aquele vetor de treinamento. Os parâmetros da rede são ajustados e a saída é comparada com

a saída desejada originando um erro. Em outras palavras, pode-se basicamente dizer que: o conhecimento do "professor" foi transferido para a rede. Quando esse conhecimento é transferido, pode-se dispensar o "professor" e a rede interagi sozinha com o ambiente no qual ela foi treinada.

- **Não Supervisionado** - Nesse tipo de aprendizado não temos o "professor externo". A rede trabalha as entradas e se organiza de modo a classificá-las mediante algum critério de semelhança. Esse tipo de rede utiliza os neurônios como classificadores, e os dados de entrada como elementos de classificação.

4.3.6 Processo de Treinamento

O treinamento pode ser definido como um processo no qual os parâmetros livres da rede neural são adaptados por meio de um processo contínuo de estimulação do ambiente em que a rede se encontra. O tipo de treinamento é determinado pela maneira em que os parâmetros variam.

Essa definição implica na seguinte seqüência de eventos: [7]

1- A rede neural é estimulada por um ambiente.

2- A rede neural varia seus parâmetros como resultado desses estímulos.

3- A rede neural responde de uma nova maneira ao ambiente, devido as variações que tem ocorrido em sua estrutura interna.

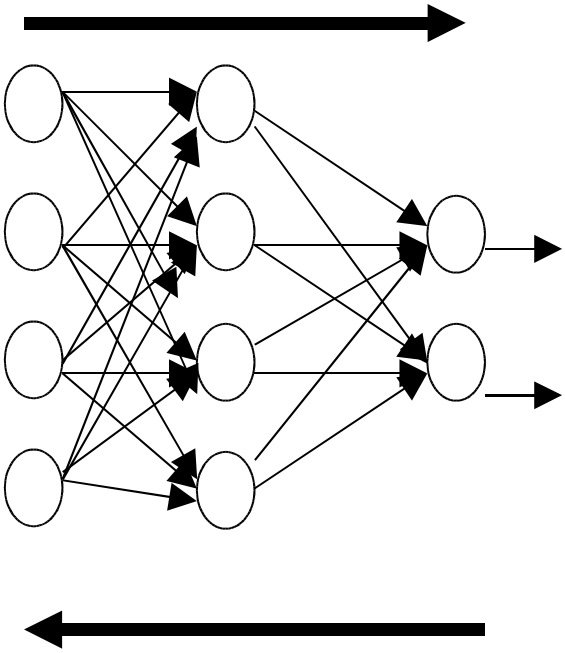
Existem vários algoritmos do tipo supervisionado para treinar redes MLP. De acordo com o modo de atualização dos parâmetros, os algoritmos de treinamento de redes do tipo MLP podem ser classificadas como: [15]

- Estáticos;
- Dinâmicos.

Enquanto os algoritmos estáticos não alteram a estrutura da rede, variando apenas os valores de seus pesos, os algoritmos dinâmicos podem tanto reduzir quanto aumentar o tamanho da rede.

O algoritmo de aprendizado mais conhecido para treinamento das redes MLP's é o algoritmo *Backpropagation*. Este algoritmo é supervisionado e seu treinamento ocorre em duas fases, onde cada fase percorre a rede em um sentido. Essas duas fases são chamadas de: *Forward* e *Backward*. A fase *Forward* é utilizada para definir a saída da rede para um dado padrão de entrada, nenhuma alteração nos pesos é feita. Na fase *Backward* são utilizadas a saída desejada e a fornecida pela rede para que os pesos sejam atualizados.[7]

Fase *forward*



Fase *Backward*

FIGURA 4.4: Fluxo do processamento do algoritmo *Back-propagation*

4.3.7 Dinâmica de Treinamento [6]

A operação da rede neural constitui de três etapas:

- **treinamento** - ajuste dos parâmetros do modelo.
- **teste** - validação dos parâmetros do modelo.
- **produção** - utilização do modelo.

No treinamento é escolhido o algoritmo e os parâmetros de aprendizado. A atualização desses parâmetros pode ser feita de duas formas:

- **Incremental** - os parâmetros são ajustados somente após a apresentação de cada padrão de treinamento. Esta abordagem é estável se a taxa de aprendizado for pequena e é geralmente mais rápida, principalmente, quando o conjunto de treinamento for redundante e grande. Uma outra vantagem desta técnica é que ela requer menos memória.

- ***Batch*** - os parâmetros são ajustados após todos os padrões terem sido apresentados. Esta técnica é geralmente mais estável, mas pode ser mais lenta se o conjunto de treinamento for grande e redundante. Esta abordagem apresenta uma estimativa mais precisa do vetor gradiente, ao custo da necessidade da mais memória.

4.3.8 Superfície de Erro [15]

A superfície de erro obtida por meio da equação 4.14, dependendo do tipo de unidade de processamento utilizada para construir a rede, pode identificar duas situações diferentes:

- A rede ser formada inteiramente por unidades de processamento lineares, nesse caso a superfície de erro é dada exatamente pela função quadrática dos pesos da rede e terá um único mínimo que será o global.
- A rede ser formada por unidades de processamento não-lineares. Neste caso, a superfície do erro poderá ter, além do mínimo global, um ou mais mínimos locais.

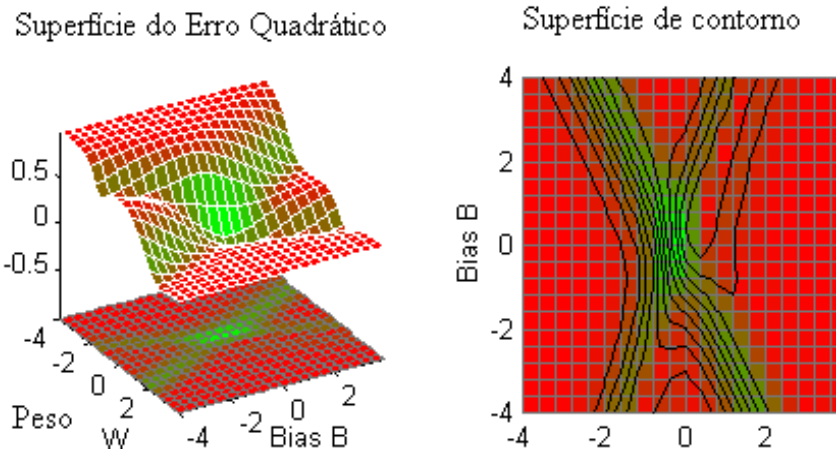


FIGURA 4.5: Superfície de erro gerada por unidades de processamento não lineares.

4.3.9 Correção do Erro no Treinamento [18]

Seja d_k (n) a resposta desejada para o neurônio k no tempo n e y_k (n) a resposta atual desse neurônio produzida pelo estímulo, x (n), aplicado na entrada da rede na qual o neurônio k está localizado, pode-se definir o sinal de erro como a diferença entre a resposta desejada e a atual, isto é

$$e_k(n) = d_k(n) - y_k(n) \tag{4.14}$$

O propósito principal da correção do erro no treinamento é minimizar a função custo baseada no sinal de erro, e_k (n), tal que a resposta atual de cada saída aproxime-se da resposta desejada para aquele neurônio.

Um critério comumente utilizado para a função custo é o critério do erro médio quadrático, definido como o valor médio quadrático da soma dos erros quadráticos:

$$E = \frac{1}{2}\sum_{p}\sum_{i=1}^{k}\left(d_i^p - y_i^p\right)^2 \qquad (4.15)$$

onde E é a medida do erro total, p é o número total de padrões, k é o número de unidades de saída, d_i é i-ésima saída desejada e y_i é a i-ésima saída gerada pela rede. O fator 1/2 é utilizado para simplificar os cálculos em possíveis derivações que são resultantes de minimizações de E com respeito aos parâmetros livres da rede. A minimização da função custo com respeito aos parâmetros livres da rede guia-nos para um método chamado de gradiente descendente .

4.3.10 Parâmetros Utilizados no Treinamento

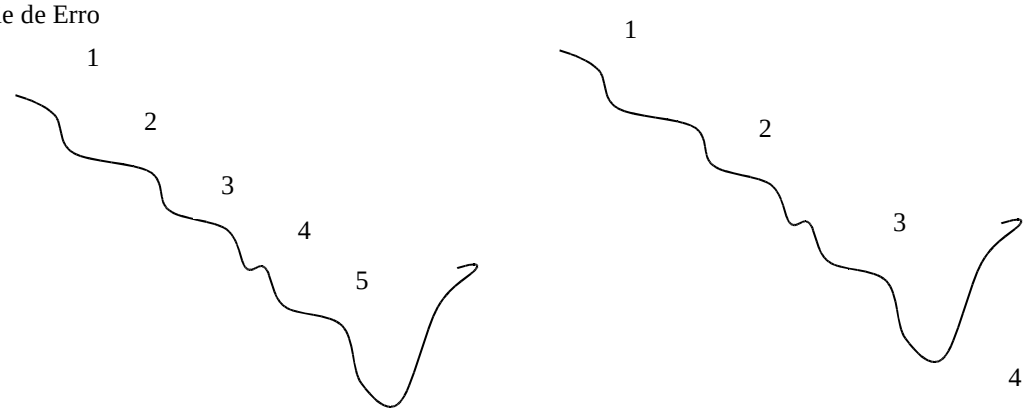
a) Momento [15, 7]

O momento é introduzido no treinamento com o objetivo de acelerar o aprendizado sem causar oscilação. Possibilita a rede ignorar as variações de alta freqüência na superfície de erro, diminuindo a probabilidade do processo de convergência parar em um mínimo local. A introdução do momento consiste em fazer com que as mudanças nos pesos das conexões sejam iguais à soma de uma fração da última alteração nestes pesos com a nova alteração determinada pela regra de aprendizagem. Dessa maneira, se a alteração anterior for realizada no sentido descendente da superfície de erro, parte da alteração atual nos pesos das conexões será realizada naquele mesmo sentido, isto é,

$$\Delta W_{i,j}(K+1) = -m\Delta W_{i,j}(k) + (1-m)\eta\delta_{pi}a_{pj} \qquad (4.16)$$

onde m é o momento, n a taxa de aprendizado adaptativa, a_{pj} é a ativação da j-ésima entrada para o p-ésimo padrão apresentado e δ_{pi} é a função de erro.

Superfície de Erro



6

Sem momento Com momento

FIGURA 4.6: Influência do termo momento na velocidade de treinamento da rede MLP

b) Taxa de Aptrendizagem [6]

A taxa de aprendizagem influencia a magnitude dos pesos, isto é, uma pequena taxa de aprendizagem implica em pequenas variações, tornando o treinamento lento e aumentando a probabilidade de paradas em mínimo local, entretanto, ao utilizar altas taxas de treinamento a rede neural poderá saturar ou até mesmo oscilar. Uma alternativa é utilizar a taxa de treinamento adaptativa, isto é, quando o erro aumentar o valor da taxa de aprendizagem diminuirá rapidamente, por outro lado se o erro diminuir a taxa de aprendizagem aumentará lentamente.

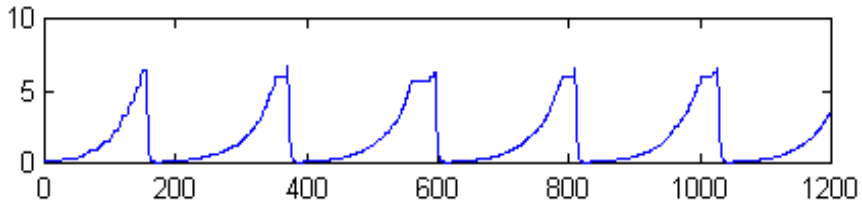


FIGURA 4.7: Taxa de aprendizagem adaptativa

4.4 DERIVAÇÃO DAS FÓRMULAS DO ALGORITMO BACKPROPAGATION [7, 15, 17]

Embora o erro total E seja definido pela soma dos nodos de saída para todos os padrões, supõe-se, sem perda de generalidade, que a minimização do erro para cada padrão individualmente levará a minimização do erro total. Assim o erro passa a ser definido pela Equação 4.16:

$$E = \frac{1}{2}\sum_{j=1}^{k}\left(d_j - y_j\right)^2 \qquad (4.17)$$

A regra delta sugere que a variação dos pesos seja definida de acordo com o gradiente descendente, isto é, de acordo com a equação 4.17:

$$\Delta w_{ji} \alpha - \frac{\partial \mathrm{E}}{\partial \mathrm{w}_{\mathrm{ji}}} \tag{4.18}$$

Utilizando a regra da cadeia, tem-se que:

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial net_j} \frac{\partial net_j}{\partial w_{ji}} \tag{4.19}$$

como $net_j = \sum_{i=1}^{n} x_i w_{ji}$. A segunda derivada , $\frac{\partial net_j}{\partial w_{ji}}$, é igual a:

$$\frac{\partial net_j}{\partial w_{ji}} = \frac{\partial \sum_{l=1}^{n} x_l w_{jl}}{\partial w_{jl}} = x_i \tag{4.20}$$

A primeira derivada localizada à direita da Equação 4.19 mede o erro no nodo j e o cálculo desta derivada também pode ser definida pela regra da cadeia:

$$\delta_j = \frac{\partial E}{\partial net_j} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial net_j} \tag{4.21}$$

A segunda derivada da Equação 4.21 é definida como:

$$\frac{\partial y_j}{\partial net_j} = \frac{\partial f\left(net_j\right)}{\partial net_j} = f'\left(net_j\right) \tag{4.22}$$

Já a primeira derivada vai depender da camada onde o nodo j se encontra. Se o nodo j estiver na última camada, o seu erro pode ser definido utilizando-se a equação 4.17.

$$\frac{\partial E}{\partial y_j} = \frac{\partial \left(\frac{1}{2} \sum_{i=1}^{k} \left(d_i - y_i\right)^2 \right)}{\partial y_j} = -\left(d_j - y_j\right) \tag{4.23}$$

Substituindo as Equações 4.23 e 4.22 em 4.21 tem-se que:

$$\delta_j = -\left(d_j - y_j\right) f'\left(net_j\right) \tag{4.24}$$

Substituindo 4.24 e 4.20 em 4.18 tem-se que:

$$\Delta w_{ij} = x_i \left(d_j - y_j\right) f'\left(net_j\right) \tag{4.25}$$

Se o nodo j não estiver na camada de saída tem-se que:

$$\frac{\partial E}{\partial y_j} = \sum_{l=1}^{M} \frac{\partial E}{\partial net_l} \frac{\partial net_l}{\partial y_j} = \sum_{l=1}^{M} \frac{\partial E}{\partial net_l} \frac{\partial \sum_{i=1}^{n} w_{il} y_i}{\partial y_j} = \sum_{l=1}^{M} \frac{\partial E}{\partial net_l} w_{jl} \tag{4.26}$$

onde

$$\sum_{l=1}^{M} \frac{\partial E}{\partial net_j} w_{jl} = \sum_{l=1}^{M} \delta_l w_{jl} \tag{4.27}$$

Substituindo as Equações 4.26 e 4.22 em 4.21 tem-se que:

$$\partial_j = f'\left(net_j\right) \sum_l \delta_l w_{lj} \tag{4.28}$$

Pode-se então generalizar a Equação 4.17 para:

$$\Delta w_{ji} = \eta \delta_j x_i \tag{4.29}$$

ou

$$w_{ji}(t+1) = w_{ji}(t) + \eta \delta_j(t) x_i(t) \tag{4.30}$$

O processo de aprendizado pode ser entendido como uma combinação de pesos e limiares que irão correspondem a um ponto na superfície de erro. Considerando que a altura de um ponto é diretamente proporcional ao erro associado a este ponto, a solução está nos pontos mais baixos da superfície.

Uma dúvida que surge naturalmente diz respeito a quando parar o treinamento da rede. Existem vários métodos para a determinação do momento onde o treinamento deve ser encerrado, entre eles pode-se citar:

- Encerrar o treinamento após M ciclos;
- Encerrar o treinamento após o erro quadrático médio ficar abaixo de uma constante;
- Encerrar o treinamento quando a porcentagem de classificações corretas estiver acima de uma constante;
- Encerrar o treinamento quando o erro médio quadrático não diminuir durante N ciclos;
- Combinação dos métodos acima.

4.5 DIFICULDADES ENCONTRADAS NO TREINAMENTO DA REDE MLP UTILIZANDO O ALGORITMO *BACKPROPAGATION* [7,15]

Um dos problemas enfrentados no treinamento de redes MLP diz respeito à definição de seus parâmetros. Pequenas diferenças nesses parâmetros poderão recolocar o ponto inicial em um local na superfície de erro, que poderá impedir que o erro desejado seja alcançado. Uma possível solução para este problema é treinar a rede várias vezes iniciando de um ponto diferente. Após os treinamentos terem sido realizados, será

selecionado o modelo que originou o menor erro. Um outro problema que poderá ocorrer é o *overfitting*, isto é, a rede memorizou os dados de treinamento tornando-se muitas vezes incapaz de corretamente classificar novos dados. Este problema é solucionado da seguinte forma: São selecionadas duas bases de dados que originam inicialmente saídas diferentes que tendem a igualdade durante o treinamento. Entretanto, a primeira corrigirá os pesos na fase *Backward* e a segunda se utilizará dos pesos corrigidos sem ter a função de atualizá-los. Durante M ciclos, onde M varia de acordo com o pesquisador, acompanha-se o erro com relação a segunda base de dados, se o erro permanecer constante ou aumentar durante o treinamento, o mesmo é reinicializado em um ponto diferente na superfície de erro.

Um terceiro problema do algoritmo *BackPropagation* acontece quando a derivada da função sigmoidal de uma unidade aproxima-se de zero durante o treinamento. A derivada da função sigmoidal é igual $o_j(1-o_j)$. Quando a saída da unidade se aproxima de zero ou um, a derivada da função sigmoidal se aproxima de zero. Como ajuste de pesos utiliza o valor da derivada, a unidade pode não ter seus pesos ajustados ou ajustá-los com um valor muito pequeno. Se no treinamento a rede começar a trabalhar na região de corte ou de saturação da função de transferência, dificilmente conseguirá sair dessa situação. Para amenizar este problema são propostas algumas sugestões:

- Utilizar uma medida de erro que tenda para infinito quando a derivada da sigmoidal tender para zero.
- Adicionar uma constante à derivada, evitando que o erro não seja igual a zero.
- Utilizar uma função de erro não linear.

4.6 MODELO RADIAL BASIS [7, 15]

A função de ativação na maioria das redes multi-camadas utiliza como argumento o produto escalar do vetor de entrada e do vetor de pesos neste nodo. Existem, porém, redes multi-camadas que utilizam uma função de distância entre seus vetores de entrada e seus pesos como a função de ativação. Uma destas redes é a rede de Funções Base Radiais, RBF.

4.6.1 Arquitetura [6]

Cada camada de uma rede RBF desempenha uma papel específico para o seu comportamento. Em uma rede de duas camadas, a primeira camada cujos nodos utilizam funções de bases radiais, agrupa os dados de entrada em *clusters*. Esta camada transforma um conjunto de padrões de entrada não linearmente separável em um conjunto de saída linearmente separável. A segunda camada, camada de saída, procura classificar os padrões

recebidos da camada anterior. As funções radiais são uma classe especial de funções cujo valor diminui ou aumenta em relação à distância de um ponto central.

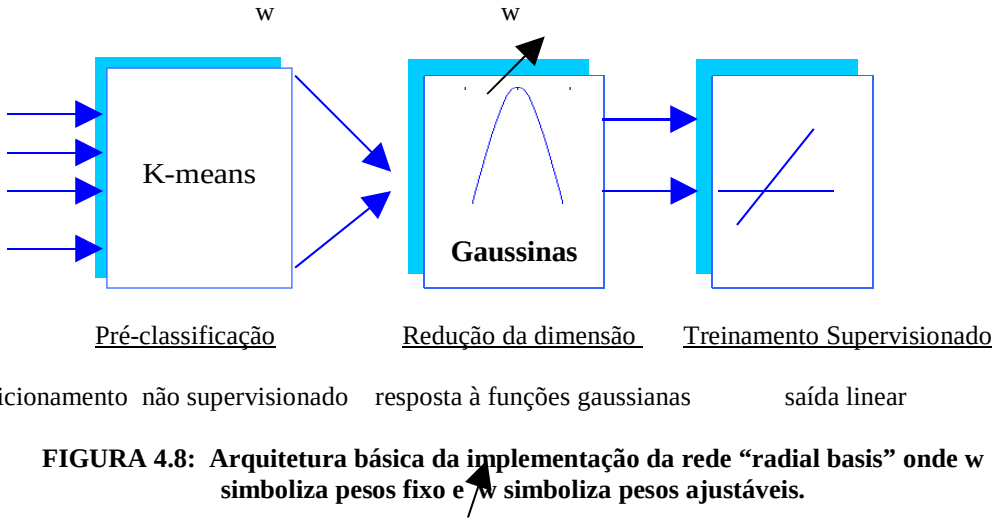


FIGURA 4.8: Arquitetura básica da implementação da rede "radial basis" onde w simboliza pesos fixo e w simboliza pesos ajustáveis.

4.6.2 Funções de Base Radias [15]

Durante o projeto de uma rede RBF é necessário definir o número de nodos da camada intermediária. Como cada nodo agrupa os padrões em um *cluster*, que é posteriormente utilizado pelos nodos da camada de saída, esta escolha deve ser feita de forma rigorosa.

Uma alternativa é definir o número de nodos de acordo com o número de padrões de entrada. Com isso, a rede RBF mapeia com exatidão do vetor de entrada para a saída correta. Contudo, esta interpolação exata é indesejável, principalmente no caso de exemplos com ruído, pois pode levar a rede ao *overfitting*.

Uma das maneiras de tratar este problema é a utilização de um número de *clusters* que seja inferior ao número de padrões de entrada.

4.6.3 Treinamento [15]

No primeiro estágio, o número de funções radiais e seus parâmetros são determinados por métodos não supervisionados. Pode-se realizar este treinamento de acordo com algumas abordagens. A primeira abordagem seleciona os centros aleatoriamente a partir dos padrões de treinamento. Esta abordagem assume que os padrões de treinamento estão distribuídos de uma maneira representativa, isto é, os padrões são capazes de capturar as principais características do problema. Uma segunda abordagem sugere a utilização de técnicas de *clustering*. Neste caso, através de um aprendizado não supervisionado, os centros são estrategicamente posicionados em regiões de espaço onde estão situados os vetores de entrada mais representativos. O segundo estágio de treinamento ajusta os pesos do nodo de saída de modo que o problema torne-se linearmente separável.

4.7 COMPARAÇÃO ENTRE A REDE RBF E A MLP [28]

Tanto a RBF quanto a MLP são exemplos de redes neuronais não lineares e aproximadores universais. Contudo, estas duas redes diferem uma da outra em alguns importantes aspectos, como:

1. A rede RBF utiliza a distância Euclidiana entre o vetor de entrada e o centro de cada unidade na camada intermediária como função de ativação, enquanto que a MLP faz o cálculo utilizando o produto interno do vetor de entrada e o vetor sináptico daquela unidade.
2. O tempo de treinamento da MLP é na maioria das vezes maior que o da RBF.
3. Ambas as redes estimam a probabilidade Bayesiana a posteriori.
4. A rede MLP constrói aproximadores globais para mapas de entrada-saída não lineares. Consequentemente, a MLP é capaz da generalização em regiões do espaço de entrada onde pouco ou nenhum dado de treinamento está disponível. Por outro lado, a RBF usando não-linearidade local com decaimento exponencial, como é o caso da função de Gauss, constrói aproximações locais para os mapas de entrada-saída não lineares. Neste sentido, as RBF tem sensibilidade reduzida com respeito a ordem de apresentação dos dados.

CAPÍTULO V

SISTEMAS IMPLEMENTADOS

Este capítulo descreve a base de dados e as características utilizadas no treinamento dos sistemas. Também são feitas considerações sobre variações no método de treinamento do HMM e da RNA.

5.1 BASE DE DADOS

As elocuções utilizadas foram gravadas por meio da placa *Sound Blaster* 16, da *Creative Labs*, com taxa de amostragem 11025 Hz e quantização de 16 bits.

Neste trabalho foram utilizadas 1400 locuções encontradas em [6] que constam de 10 palavras pronunciadas por 68 locutores masculinos distintos. As palavras são: liga, grave, pausa, avance, siga, desliga, volte, ejete e apague. Além dessas gravações foram utilizadas 288 gravações, pronunciadas por 48 locutores masculinos distintos, em [1], que constam das palavras: pare, grave, pausa, apague e volte. Estas gravações foram realizadas com um microfone diferente do utilizado em [6], causando uma variação do meio, o que veio a testar a generalização dos sistemas apresentados na tese.

As 1688 locuções foram divididas em três grupos:

- No grupo 1 foram utilizadas as locuções gravadas em [6]. Estas locuções foram divididas em dois subgrupos: treinamento e teste. No treinamento foram utilizadas 1000 locuções e no teste foram usadas 400 locuções.
- No grupo 2 foram utilizadas as locuções gravadas em [6]. No treinamento foram utilizadas 600 das 1000 locuções do grupo 1, sendo 60 repetições de cada palavra e no teste as 220 locuções do grupo 1, referidas no item anterior.

- No grupo 3 foram utilizadas 60 locuções restantes do grupo 1 somadas às 391 locuções encontradas em [1].

5.2 CARACTERÍSTICAS UTILIZADAS PELO HMM E PELA RNA

As características utilizadas pelo HMM encontram-se na Tabela 3.2. As da RNA foram as seguintes:

- Relação da taxa de cruzamentos de zeros entre as duas metades do sinal
- 2º *Mel Cepstrum*
- 1º *Cepstrum*
- Faixa de Energia
- 3º *Mel Cepstrum*

5.3 CONSIDERAÇÕES SOBRE O HMM

O desempenho do HMM depende da escolha do modelo e dos parâmetros iniciais de treinamento, assim foram analisados alguns dos seus parâmetros, quais sejam:

- Tipo de inicialização do treinamento
- Número de estados
- Número de misturas por estado
- Modelo de *Bakis*
- Fator de Convergência

Os treinamentos utilizaram o algoritmo *Segmental K-means* e os modelos estão na Tabela 5.1.

TABELA 5.1: Modelos utilizados no estudo do HMM

Modelos	
6 estados e 15 misturas (6e15m)	20 estados e 15 misturas (20e15m)
7 estados e 15 misturas (7e15m)	8 estados e 5 misturas (8e5m)
8 estados e 15 misturas (8e15m)	8 estados e 11 misturas (8e11m)
9 estados e 15 misturas (9e15m)	8 estados e 14 misturas (8e14m)
10 estados e 15 misturas (10e15m)	8 estados e 17 misturas (8e17m)
11 estados e 15 misturas (11e15m)	8 estados e 20 misturas (8e20m)
12 estados e 15 misturas (12e15m)	8 estados e 23 misturas (8e23m)
13 estados e 15 misturas (13e15m)	8 estados e 28 misturas (8e28m)

5.3.1 Tipo de Inicialização do Treinamento

Os parâmetros iniciais do HMMs obedeceram às imposições acústicas das locuções e às imposições estatísticas do modelo, como é mostrado a seguir:

1) probabilidade inicial: fixa em todos os treinamentos. De acordo com o modelo de *Bakis*,

$$\pi_1 = 0 \quad \text{e} \quad \pi_n = 0 \text{ para } n \neq 1$$

2) valores da matriz inicial das probabilidades das transições: aleatórios, respeitando-se as restrições do modelo.

3) valores da matriz inicial da densidade de probabilidade de saída das observações: visando uma melhor estimativa dos valores iniciais, foi seguido o seguinte procedimento:

a) Segmentação das R sequencias de observações nos N estados, sendo resto da divisão somado ao último estado.

b) Em cada estado utilizou-se o algoritmo *Kmeans-modificado* para separar as observações em M grupos e obtiveram-se valores das probabilidades de observações por meio da mistura de M Gaussianas.

5.3.2 Número de Estados

De acordo com a Tabela 5.2, de 9 a 14 estados é o suficiente para modelar palavras isoladas, quando são utilizadas 15 misturas.

TABELA 5.2: Desempenho do HMM de acordo com o número de estados

Modelo	Taxa de Reconhecimento	
	Grupo 1	Grupo 3
6e15m	97,46 %	72,97 %
7e15m	88,35 %	75,31 %
8e15m	90,6 %	75,36 %
9e15m	95,3 %	75,95 %
10e15m	95,45 %	77,0 %
11e15m	96,70 %	77,43 %
12e15m	96,96 %	78,82 %
13e15m	95,69 %	76,04 %
14e15m	96,70 %	78,82 %

De acordo com a Tabela 5.2 foi traçado o gráfico da Figura 5.1 referente ao grupo 1.

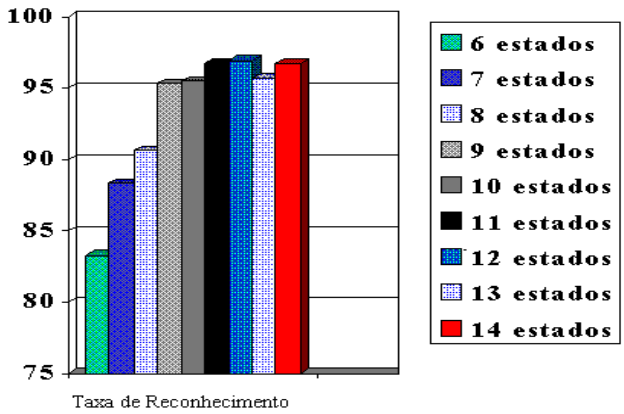


FIGURA 5.1: Taxa de Reconhecimento de acordo com o número de estados

5.3.3 Número de Misturas por Estado

Uma mesma palavra raramente é pronunciada da mesma forma por um mesmo locutor; assim a função de densidade acústica torna-se muito complexa e com uma única mistura não se consegue modelá-la. Na Tabela 5.3 é mostrada a taxa de reconhecimento obtida para modelos com um número diferente de misturas.

TABELA 5.3: Desempenho do HMM de acordo com o número de misturas

Modelo	Taxa de Reconhecimento	
	Grupo 1	Grupo 3
8e5m	95,95 %	70,38%
8e11m	98,53 %	71,32%
8e14m	98,22 %	70,73%
8e17m	98,04 %	69,69%
8e20m	97,97 %	69,34%
8e23m	97,97 %	70,03%
8e28m	98,22 %	71,08%

De acordo com a Tabela 5.3 é traçado o gráfico da Figura 5.2.

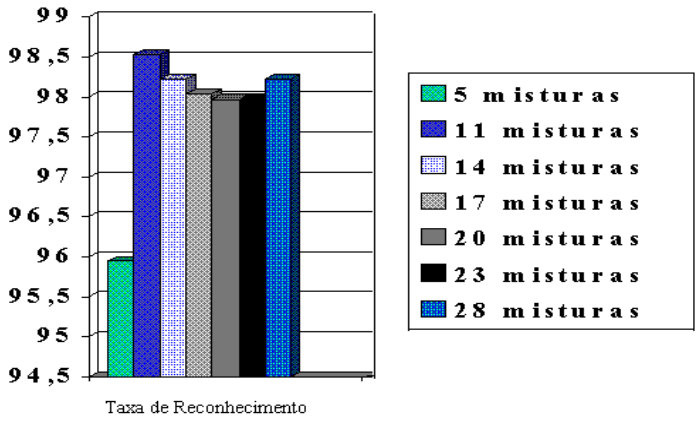


FIGURA 5.2: Taxa de Reconhecimento de acordo com o número de misturas

5.3.4 Modelo de *Bakis*

Segundo [33], o modelo de *Bakis* é o recomendado quando se quer modelar sinais de voz. Este item mostra os resultados alcançados para um modelo com delta igual a 3 e para o modelo de *Bakis*. De acordo com a Tabela 5.4, o desempenho do modelo de *Bakis* mostrou-se superior ao do modelo proposto.

TABELA 5.4: Taxa de Reconhecimento

MODELO	Taxa de Reconhecimento			
	Modelo de *Bakis*		Modelo Proposto	
	Grupo 1	Grupo 3	Grupo 1	Grupo 3
6e15m	97,46%	74,97%	97,30%	72,15%
7e15m	88,35 %	75,31 %	88,73%	73,52%
8e5m	95,95%	70,38%	93,14%	59,58%

5.3.5 Fator de Convergência

O fator de convergência é um parâmetro que influencia a proximidade entre a distribuição acústica real da distribuição ideal, isto é, quanto menor o fator de convergência menor será a distância entre as distribuições. Os modelos deste trabalho utilizaram um fator de convergência igual a 1%. Para uma análise deste parâmetro, foram treinados dois modelos com um fator de convergência de 0.1%. Os resultados estão na Tabela 5.5.

TABELA 5.5: Taxa de Reconhecimento de acordo com o fator de convergência

MODELO	Taxa de Reconhecimento	
	Modelo com fator igual a 1%	Modelo com fator igual a 0.1%

	Grupo 1	Grupo3	Grupo 1	Grupo 3
8e14m	98,22%	70,73%	98,03%	69,69%
8e11m	98,53%	71,2%	97,79%	70,73%

5.3.6 Conclusões sobre o HMM

Analisando os resultados conclui-se que: os parâmetros iniciais influenciam diretamente o desempenho do modelo. Como existe uma infinidade de combinações de parâmetros, o modelo ideal dificilmente será alcançado. Dessa forma, o objetivo do estudo do HMM foi de procurar um modelo de menor estrutura com um desempenho satisfatório em vez de procurar o melhor modelo.

5.4 CONSIDERAÇÕES SOBRE A RNA

O estudo sobre a RNA enfocou diversos itens, quais sejam:

- Parâmetros de uma Rede MLP.
- Normalização dos pesos.
- Critério de Parada do treinamento da RNA.
- Taxa de Aprendizagem e momento.
- Variações no treinamento da rede MLP.

Assim como no HMM, o desempenho da Rede Neural está relacionado com a definição de seus parâmetros.

5.4.1 Parâmetros de uma MLP

As escolhas do número de camadas, do número de neurônios por camada, das funções de transferências em cada camada, da parada do treinamento não seguem regras fixas.

A escolha de muitas camadas intermediárias leva a memorização dos dados treinados e o treinamento torna-se ainda mais lento como foi visto na prática. Assim, procura-se utilizar a menor estrutura possível para classificar corretamente os padrões.

Para afirmar que um conjunto de pesos é o melhor para uma determinada estrutura, a rede deve ser treinada várias vezes com pesos iniciais diferentes. Entretanto, essa técnica torna-se impraticável, quando se faz uma projeção para uma das redes encontradas neste trabalho, a qual possui três camadas e cujo treinamento prolongou-se durante 2 semanas. Se este treinamento fosse repetido 5 vezes, o tempo aumentaria para 10

semanas e ainda, provavelmente, não seria obtido o conjunto de pesos ideal. Como na literatura se destacam 3 funções de transferência e foram utilizadas 3 camadas, existem 27 combinações possíveis entre camadas e funções, em consequencia o tempo de treinamento aumentaria para 270 semanas.

Como pode ser visto, a escolha dos parâmetros que compõem uma rede é feita de modo mais ou menos empírico.

5.4.2 Normalização dos Pesos

O treinamento de uma rede MLP deve ser feito na faixa linear do neurônio, uma vez que não é garantido que a rede consiga sair de uma região de saturação ou de corte. Para comprovar a dificuldade em se trabalhar na região de saturação, foi escolhida uma matriz de entrada aleatória com valores entre 0 e 1 de dimensões 20×50 e duas matrizes de pesos com valores entre 50 e 500. Como a dimensão da matriz de entrada é reduzida, o treinamento pôde ser repetido 20 vezes. De acordo com a Tabela 5.6, pode ser visto que, ao normalizar, os pesos o treinamento torna-se mais rápido.

TABELA 5.6: Número de *epochs* médio para pesos normalizados e não normalizados

MODELO	Número de *Epochs* Médio	
	Com Normalização	Sem Normalização
RNA1 10×2	721.2	4000

5.4.3 Critério de Parada do Treinamento da RNA

De acordo com [15], o treinamento pode ser interrompido quando:

- É atingido o número máximo de *epochs* durante o treinamento.
- É atingido o erro minimo quadrático desejado.
- É atingida uma determinada porcentagem dos dados de treinamento corretamente classificados.
- Combinação de todos os critérios anteriores.

Neste trabalho utilizou-se a combinação dos três primeiros critérios, isto é, foram escolhidos um número máximo de *epochs* e o erro mínimo, ficando a porcentagem dos dados corretamente treinados definida da seguinte forma: quando um elemento do vetor de saída encontra-se entre 0.95 e 1.05 é aproximado para 1, de outra forma para 0. No final de uma ciclo, a rede gera uma matriz de saída que é aproximada e subtraída da matriz de saída ideal, originando uma matriz de erros. Logo em seguida é feita uma soma em cada coluna da matriz de erros. Se a soma, na coluna, for igual a 0 o padrão foi corretamente classificado.

Durante o treinamento da rede foram utilizadas a matriz aleatória e a estrutura encontrada no item 5.4.2. Na Figura 5.3 é mostrado o erro médio quadrático obtido durante o treinamento e na Figura 5.4 a porcentagem dos dados corretamente treinados.

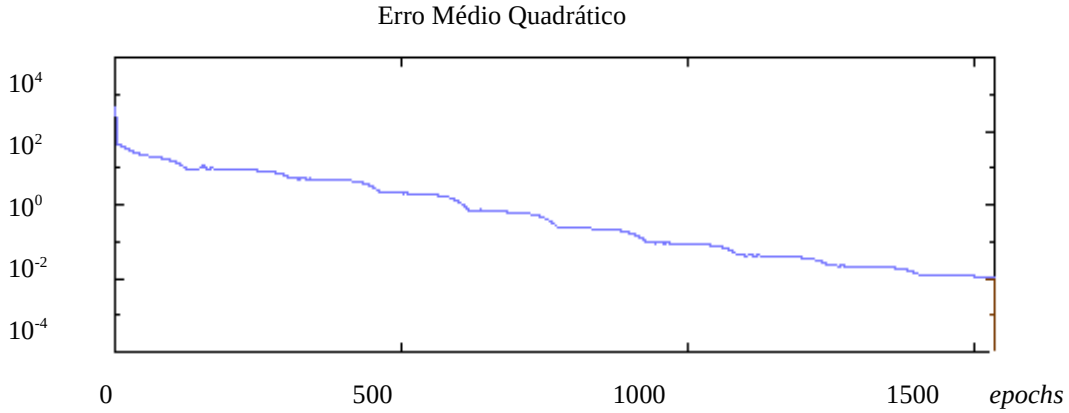


FIGURA 5.3: Controle do erro médio quadrático durante o treinamento

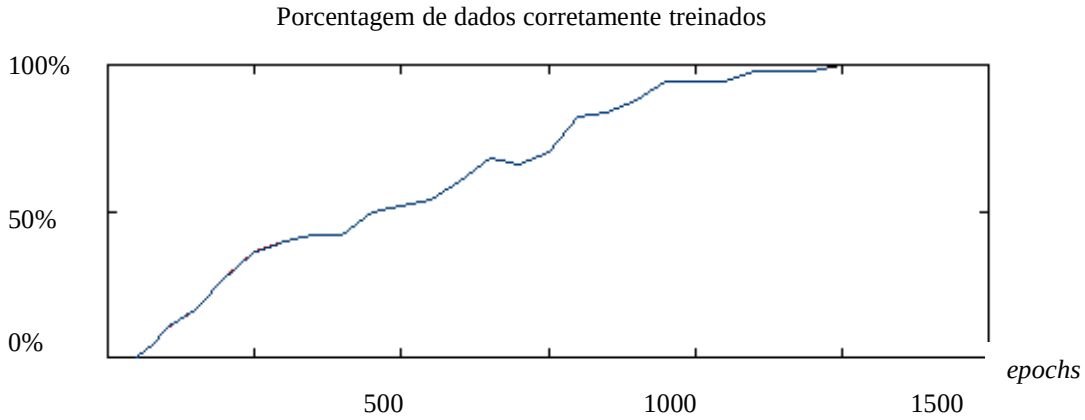


FIGURA 5.4: Controle dos dados corretamente classificados durante o treinamento

Com este critério de treinamento, torna-se mais fácil analisar os resultados da rede.

5.4.4 Taxa de Aprendizagem e Momento

A taxa de aprendizagem pode ser constante ou adaptativa e o momento é um parâmetro que acelera o treinamento da rede. Na Tabela 5.7 é mostrado o número de *epochs* médio para uma rede que utiliza taxa de aprendizagem adaptativa e momento e outra que utiliza taxa de aprendizagem constante. O treinamento foi repetido 20 vezes e o número máximo de *epochs* foi 4000.

TABELA 5.7: Média de *epochs*

MODELO	Média de *Epochs*	
	Taxa Aprendizagem Adaptativa com Momento	Taxa de Apredizagem Constante
RNA1 10×2	906.9	4000

De acordo com a Tabela 5.7, a rede deverá ser treinada com taxa de aprendizagem adaptativa com momento.

5.4.5 Variação no Treinamento da Rede MLP

Para avaliar a importância da maximização da distância entre as classes no desempenho da rede MLP, esta característica foi desconsiderada durante o treinamento. Desse modo foram treinadas duas redes, sendo a matriz de entrada composta por padrões de voz:

- RNA1 – A rede RNA1 é constituída de 10 "sub-redes" que foram treinadas separadamente, isto é, os dados de entrada de uma sub-rede só contém os padrões de uma única palavra.
- RNA2 - Assim como na rede RNA1, foram treinadas 10 "sub-redes" separadamente. Entretanto nos dados de entrada foram considerados os padrões dos dez comandos e na matriz de saída só foram "habilitados" os vetores correspondentes a palavra que se desejava treinar sendo os outros zerados. Quando o vetor é habilitado utiliza-se o código de bits da Tabela 5.8.

Na rede RNA1 não existe a maximização da distância entre as classes; na rede RNA2 foi utilizada a maximização de uma palavra com relação as outras 9. A estrutura das três redes foi $10 \times 30 \times 60$; a função de transferência utilizada nas duas primeiras camadas foi a *logsig* e na camada de saída a *purelim*. Na camada de saída da rede foi utilizado um código de bits ortogonais, quais sejam:

TABELA 5.8: CÓDIGO DE BITS ORTOGONAIS

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
***P1**	1	0	0	0	0	0	0	0	0	0
P2	0	1	0	0	0	0	0	0	0	0
P3	0	0	1	0	0	0	0	0	0	0
P4	0	0	0	1	0	0	0	0	0	0
P5	0	0	0	0	1	0	0	0	0	0
P6	0	0	0	0	0	1	0	0	0	0
P7	0	0	0	0	0	0	1	0	0	0
P8	0	0	0	0	0	0	0	1	0	0
P9	0	0	0	0	0	0	0	0	1	0
P10	0	0	0	0	0	0	0	0	0	1

*** P1 = Palavra 1**

Na fase de teste do sistema, o vetor de entrada e o vetor de saída ideal associado a ele são conhecidos. Em cada uma das dez redes foi calculado o erro médio quadrático entre o vetor gerado e o vetor de saída ideal. Após a construção da matriz de erro foi selecionado o menor erro, e a palavra associada a ele é a que foi reconhecida pela RNA. A estrutura utilizada no reconhecimento das redes RNA1 e RNA2 pode ser visto na Figura 5.5

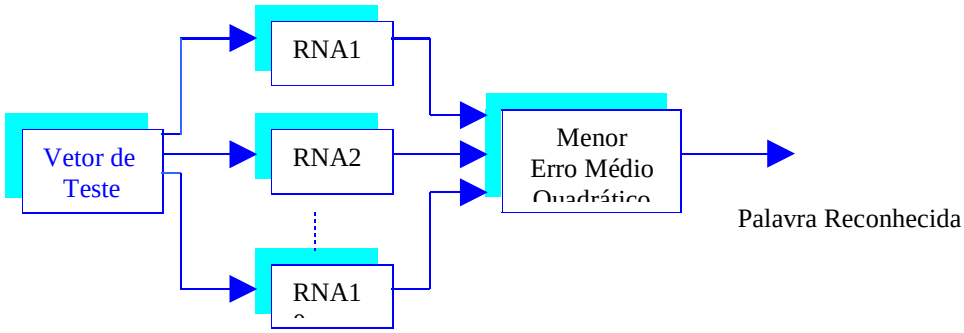


FIGURA 5.5: Reconhecimento de Palavras Isoladas

O número de "epochs" e o erro médio encontram-se na Tabela 5.9, e o desempenho na Tabela 5.10.

TABELA 5.9: Erro médio e número de *epochs* de RNA1 e RNA2

MODELO	Erro Médio	Número de *epochs*
RNA1	.001	456,2
RNA2	.001	2456,24

TABELA 5.10: Taxa de reconhecimento da RNA1 e da RNA2

MODELO	Taxa de Reconhecimento	
	GRUPO 1	GRUPO 3
RNA1	66,3 %	23,34%
RNA2	91,17%	40,77%

Das Tabelas 5.9 e 5.10 pode-se concluir que:

- Um erro médio baixo não indica que a taxa de reconhecimento seja alta.
- Desconsiderando a maximização da distância entre as classes, a taxa de reconhecimento será baixa.

5.5 REDES MLP

A matriz de entrada tem dimensões de 600 × 1000, isto porque foram utilizadas 5 características, 120 janelas e 100 repetições de cada uma das dez palavras. Já a matriz de saída é formada pelo código de bits ortogonais da Tabela 5.8.

De acordo com as consideração sobre a RNA, os parâmetros de treinamento e o critério de parada foram:

- O critério utilizado para terminar o treinamento é formado pela combinação dos seguintes itens: número máximo de *epochs*, erro mínimo quadrático e porcentagem de dados corretamente treinados.
- A taxa de aprendizagem escolhida foi a do tipo adaptativa; para acelerar o treinamento foi utilizado o momento e a matriz de entrada e pesos foram normalizados entre -1 e 1.

O erro mínimo quadrático escolhido foi 1, isto porque a matriz de saída é constituída por 1000 colunas com 10 linhas cada uma, assim cada elemento vetorial terá, em média, um erro de 0,0001 com relação ao valor ideal, e este valor foi considerado razoável para uma rede neural.

Para evitar a memorização dos dados treinados foram utilizados 90 neurônios na primeira camada escondida e 10 na camada saída, como é mostrado a seguir na Figura 5.6:

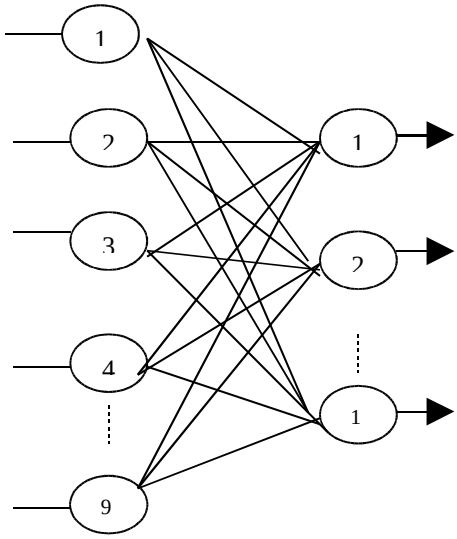


FIGURA 5.6: Arquitetura da Rede MLP

O desempenho da rede MLP é mostrado na Tabela 5.11. A escolha dos parâmetros e da arquitetura da rede foi baseada nos resultados do item 5.4.

TABELA 5.11: Taxa de reconhecimento da rede MLP

Modelo	Taxa de Reconhecimento	
	GRUPO 1	GRUPO 3
RNA3	98,5%	75,82%

5.6 - REDE RADIAL BASIS

A rede Radial Basis utilizada neste trabalho é composta de duas camadas escondidas e uma de saída, sendo a camada de entrada formada pelas funções radiais, cujo valor diminui ou aumenta em relação à distância de um ponto central. A função, utilizada foi a gaussiana, isto é,

$$f(u) = \exp\left(-\frac{v^2}{2\sigma^2}\right) \tag{5.1}$$

onde $v = \|x - \mu\|$, dado geralmente pela distância Euclidiana, x é o vetor de entrada e μ e σ representam o centro e a largura da função radial radial respectivamente. A arquitetura da rede Radial Basis está representada na Figura 5.7:

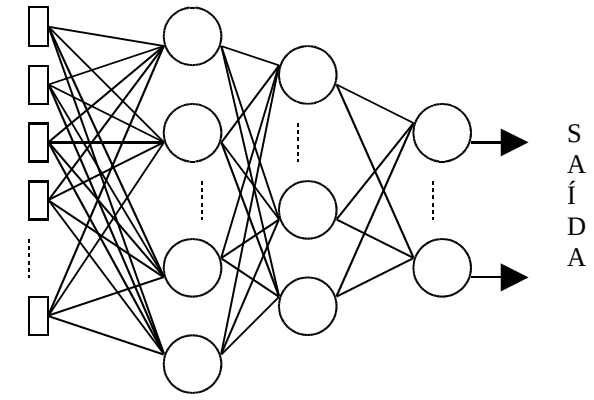


FIGURA 5.7: Rede RBF com duas camadas intermediárias

Segundo [15], é indesejado um número de funções radiais igual ao número total de classes de treinamento, principalmente no caso de exemplos com ruído, pois pode levar ao *overfitting*. Uma outra maneira consiste em utilizar um número de funções radiais menor que o número total de classes. Além disso, as posições dos centros não devem ficar restritas apenas aos vetores de entrada. Desse modo, foi feito um estudo

comparativo utilizando-se 8, 10 e 25 centros para cada uma das 5 características. A matriz de saída das três redes é formada pelo código de bits ortogonais da Tabela 5.8. A Tabela 5.12 mostra o desempenho das 3 redes.

TABELA 5.12: Desempenho da rede RBF com 8,10 e 15 centros

Número de Centros	Taxa de Reconhecimento		Número de *epochs*
	Grupo 1	Grupo 3	
8 centros	98 %	74,01	33.235
10 centros	97,7 %	80,1 %	13.236
25 centros	96,7 %	70,7 %	9.876

De acordo com a Tabela 5.12, com um número de centros igual ao número de classes obtiveram-se os melhores resultados.

5.7 CONCLUSÃO SOBRE A REDE NEURAL

Assim como no HMM, a escolha do modelo da Rede Neural baseia-se na menor estrutura que proporciona um desempenho bem satisfatório. Na Figura 5.8 é mostrado como a escolha dos parâmetros e a forma de treinamento da rede afetou o desempenho da rede para o Grupo 3.

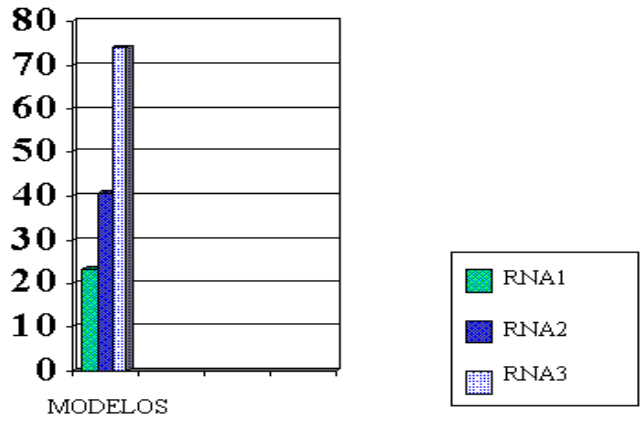


FIGURA 5.8: Taxa de reconhecimento das redes MLP

5.8 UM ESTUDO COMPARATIVO ENTRE O HMM E A REDE NEURAL

O HMM e a RNA foram comparados em três situações, quais sejam:

- Utilização de um número diferente de janelas.
- Utilização de uma quantidade diferente de dados para o treinamento.
- Utilização de uma quantidade diferente de características.

5.8.1 Utilização de um Número Diferente de Janelas

Com o número de janelas e a superposição fixos, o tamanho da janela em amostras fica definido como:[33]

$$\text{Número de Amostras por janela} = \frac{\text{Número de Amostras da elocução}}{\text{Número de Janelas} \times (1 - \text{Fração superposição})} \quad (5.2)$$

Utilizando uma taxa de amostragem de 11025 Hz, o tamanho de cada janela em segundos fica definido como:

$$\text{Tamanho da janelas em segundos} = \frac{\text{Número de Amostras por Janela}}{11025} \quad (5.3)$$

De acordo com a Equação 5.3 foi construída a Tabela 5.13

TABELA 5.13: Tempo médio de cada janela para os dez comandos

Palavras	Número Médio de Amostras	Número Médio de Amostras por Janela		Tempo Médio de cada Janela	
		120 Janelas	60 Janelas	120 Janelas	60 Janelas
LIGA	5794	202	404	18,3ms	36,6ms
PARE	4399	153	306	13,8ms	27,6ms
GRAVE	6491	226	452	20,5ms	41,0ms
PAUSA	5524	192	384	17,4ms	34,8ms
AVANCE	7391	257	514	23,3ms	46,6ms
SIGA	7361	256	512	23,2ms	46,4ms
DESLIGA	8383	291	582	26,4ms	52,8ms
VOLTE	6650	231	462	20,9ms	41,8ms
EJETE	8306	288	576	26,1ms	52,2ms
APAGUE	7948	276	552	25ms	50ms

Para avaliar a importância do tamanho da janela no RAV foram utilizados 6 modelos para o HMM e dois para a RNA, a saber:

TABELA 5.14: Modelos dos HMMs escolhidos para o desenvolvimento deste item

Modelos Utilizados	
8 estados 15 gaussianas (8e15m)	10 estados e 15 gaussianas (10e15m)
6 estados 15 gaussianas (6e15m)	12 estados e 15 gaussianas (12e15m)
9 estados e 15 gaussianas (9e15m)	13 estados e 15 gaussianas (13e15m)

TABELA 5.15: Modelos de RNA escolhidos para o desenvolvimento deste item

Modelos Utilizados	
RNA3 10×90	RNA4 10×90

Nas Tabelas 5.16 e 5.17 são mostradas as taxas de reconhecimento obtidas para os 8 modelos utilizando 60 e 120 janelas. As locuções do grupo 3 não participaram do treinamento e as locuções do grupo 1 que participaram do treinamento não fizeram parte do teste dos modelos.

TABELA 5.16: Taxa de Reconhecimento para os modelos do HMM

Modelo	60 Janelas		120 Janelas	
	Grupo1	Grupo 3	Grupo1	Grupo 3
6e15m	83,25%	74,94%	97,46%	72,97%
8e15m	90,6%	75,36%	97,46%	74,31%
9e15em	95,3%	75,95%	97,46%	72,91%
10e15m	95,45%	77,00%	97,72%	73,81%
12e15m	96,96%	78,82%	97,46%	73,96%
13e15m	95,69%	76,04%	97,72%	73,26%

TABELA 5.17: Taxa de Reconhecimento para a RNA

Modelo	Taxa de Reconhecimento	
	Grupo1	Grupo 3
RNA4	98,5%	81,82%
RNA3	98,5%	75,82%

De acordo com o item 2.2, as propriedades do sinal de voz são consideradas estacionárias entre 10 e 40ms para a língua inglesa. Entretanto, como pode ser visto nas Tabelas 5.16 e 5.17 para a língua portuguesa, que é constituída de mais sons vozeados que fricativos, esse intervalo de tempo pode ser aumentado tornando os modelos mais robustos para novos dados, além de tornar o tempo de processamento menor.

Com base nas Tabelas 5.16 e 5.17 são construídos os gráficos da Figura 5.9 e da Figura 5.10:

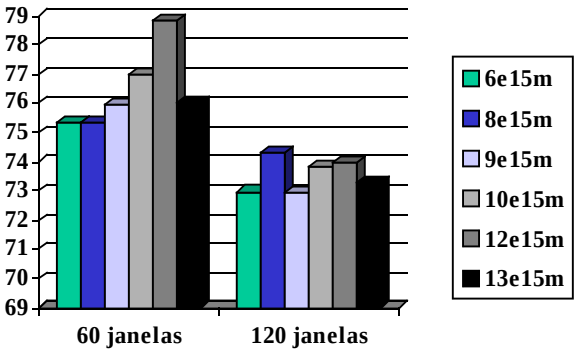


FIGURA 5.9: Desempenho do HMM de acordo com o número de janelas

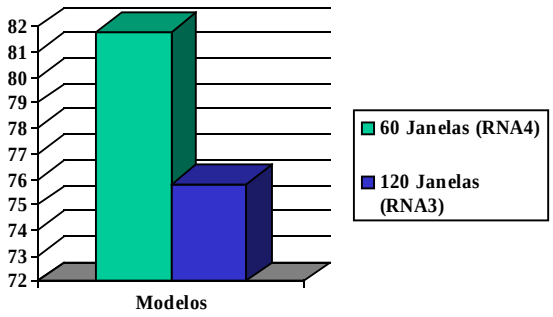


FIGURA 5.10: Desempenho da RNA de acordo com o número de janelas

5.8.2 Utilização de uma Quantidade Diferente de Dados para Treinamento

Como não foram utilizados todos os possíveis padrões acústicos e temporais para os dez comandos, alguns vetores de teste serão erroneamente classificados. Nas Tabelas 5.18 e 5.19 são mostradas as taxas de reconhecimento dos dois sistemas quando são utilizadas 100 e 60 repetições de cada comando. Com este experimento tem-se a noção de como a utilização de uma quantidade diferente de dados para o treinamento influência no desempenho dos modelos.

TABELA 5.18: Taxa de Reconhecimento do HMM para o Grupo 2

Modelo	Taxa de Reconhecimento	
	100 padrões	60 padrões
6e15m	72,97%	66,89%
8e15m	74,31%	65,62%
9e15m	72,91%	66,02%
10e15m	77,00%	71,45%

TABELA 5.19: Taxa de Reconhecimento da RNA para o Grupo 2

Modelo	Taxa de Reconhecimento	
	100 padrões	60 padrões
RNA4	98,5%	98,35%
RNA5 (8 centros)	98%	96,75%

Os resultados encontrados nas Tabelas 5.18 e 5.19 podem ser explicados da seguinte forma:

- O HMM é um modelo probabilístico, assim a utilização de uma maior quantidade de dados de treinamento, faz com que o modelo tenha uma melhor taxa de reconhecimento.
- A rede MLP não utiliza modelos probabilísticos e o seu desempenho não decairá tão drasticamente com a utilização de poucos dados de treinamento, como no HMM.

5.8.3 Utilização de uma Quantidade Diferente de Características

O HMM e a RNA foram treinados com um diferente número de características. Durante o treinamento dos modelos foi percebido que a convergência da Rede Neural é mais demorada que a do HMM. Desse modo se a RNA utilizar o mesmo número de características do HMM o seu treinamento irá tornar-se ainda mais lento ou não convergirá. Para mostrar a melhoria no desempenho do HMM quando são utilizadas mais características foram treinados 2 modelos com dois grupos de características diferentes:

1. HMM1 utilizando as características da Tabela 2.2
2. HMM2 utilizando as características: 1º Coeficiente *Cepstrum*; 1º, 2º e 3º Coeficientes *Mel Cepstrum*, e a Relação da taxa de cruzamento de zeros entre as duas metades do sinal de voz.

A Tabela 5.20 mostra o desempenho destes dois modelos.

TABELA 5.20: Taxa de Reconhecimento dos modelos com diferentes características

Modelo	Taxa de Reconhecimento	
	8 estados e 5 misturas	8 estados e 10 misturas
HMM1	92%	97,5
HMM2	85,5%	89,1%

De acordo com a Tabela 5.20 é traçado o gráfico da Figura 5.11

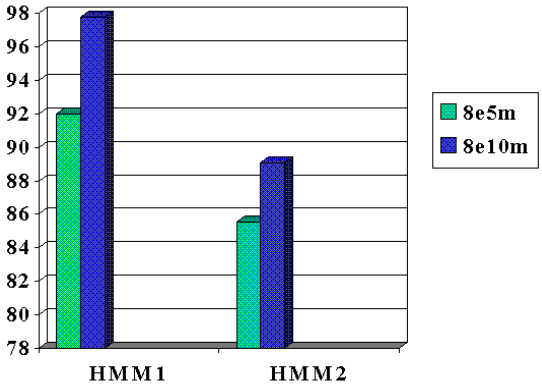


FIGURA 5.11: Taxa de Reconhecimento de acordo com o número de características.

5.9 UMA ANÁLISE COMPARATIVA

De acordo com os resultados obtidos pode-se concluir que:

- O HMM é melhor modelo para o reconhecimento de palavras isoladas, pois ainda que o seu desempenho assemelha-se ao da Rede Neural, a sua convergência foi obtida mais rapidamente. Veja a Figura 5.12.

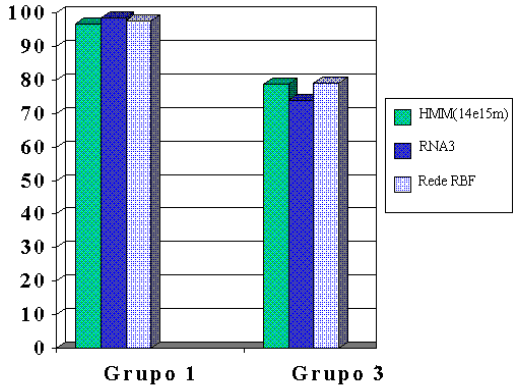


FIGURA 5.12: Análise comparativa.

- O treinamento do HMM é mais robusto com relação a quantidade de dados e de características; desse modo, ele poderá ser utilizado no reconhecimento de palavras pertencentes a um vocabulário maior. Entretanto a Rede Neural, em extensos vocabulários, somente será utilizada no pós-processamento, isto é, num sistema híbrido.
- O algoritmo de treinamento da Rede Neural é mais simples que o do HMM, entretanto o seu tempo de treinamento é maior.
- Diferentemente da Rede Neural, o HMM somente minimiza o erro médio quadrático desconsiderando a maximização da distância entre os modelos referentes a cada palavra. Desse modo, o sistema híbrido terá de ser implementado introduzindo-se nela a informação contextual.

CAPÍTULO VI

SISTEMAS HÍBRIDOS

Neste capítulo será feita uma análise dos parâmetros do HMM para que o sistema híbrido seja implementado. Após a escolha dos parâmetros serão treinados vários modelos híbridos e seus desempenhos comparados com o do HMM e da RNA.

6.1 MODELOS UTILIZADOS NO SISTEMA HÍBRIDO

O modelo escolhido para o HMM tem 6 estados e 15 misturas e o da Rede Neural duas camadas, sendo 90 neurônios na camada escondida e 10 na camada de saída. Com relação à analise do tempo foram utilizadas 60 janelas com superposição de 76%. Foram utilizadas 60 janelas porque o esforço computacional tornou-se menor e o desempenho dos dois modelos não diminui drasticamente. Os modelos foram escolhidos da seguinte forma: Procurou-se um modelo cuja a melhoria no desempenho pudesse ser notada com a implementação do sistema híbrido. Foram utilizados o número de janelas e a superposição fixos com o tamanho de cada janela variável, para diferenciar o trabalho feito em [1]. Na Tabela 6.1, tem-se o número de repetições de cada palavra.

TABELA 6.1 Número de repetições

	Grupo 1	Grupo 3
P1	74	43
P2	30	48
P3	33	48
P4	41	48
P5	34	48
P6	25	25
P7	27	42
P8	26	67
P9	35	35
P10	75	47

6.1.1 HMM

Nas Tabelas 6.2 e 6.3 encontram-se as matrizes de confusão para o Grupo 1 e o Grupo 3 do modelo 6e15m; onde na horizontal estão as palavras corretas e na vertical o número de repetições das palavras que foram avaliadas erroneamente pelo HMM.

TABELA 6.2 Matriz de Confusão do HMM para o grupo 1

	P2	P4	P5	P6	P7	P8	P10
P1				6	55		
P2							2
P3						1	
P5							1
P10	1		1				

TABELA 6.3: Matriz de Confusão do HMM para o Grupo 3

	P2	P4	P5	P6	P7	P8	P9	P10
P1				2	33		2	
P2			2			6		7
P3	5					11		4
P4			2			3		1
P5				1		5		14
P7							11	
P8		2	1					
P10					1			

Nas Tabelas 6.2 e 6.3 destacam-se 3 grupos de confusão, quais sejam:

1º grupo: formado pelas palavras 1, 6 e 7.

2º grupo: formado pelas palavras 2, 5 e 10.

3º grupo: formado pelas palavras 3 e 8.

Nas Figuras 6.1 e 6.2 estão a taxa de reconhecimento para as palavras do grupo 1 e 3.

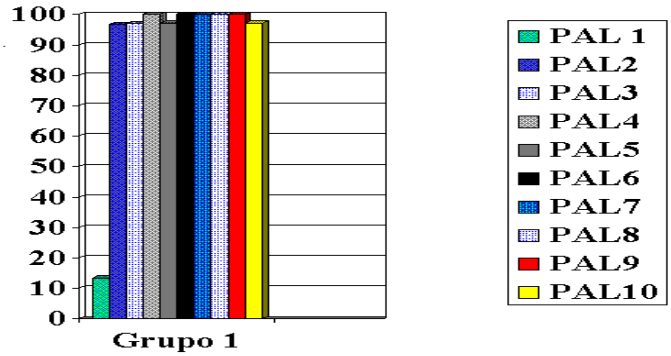


FIGURA 6.1: Taxa de Reconhecimento do HMM para o Grupo 1

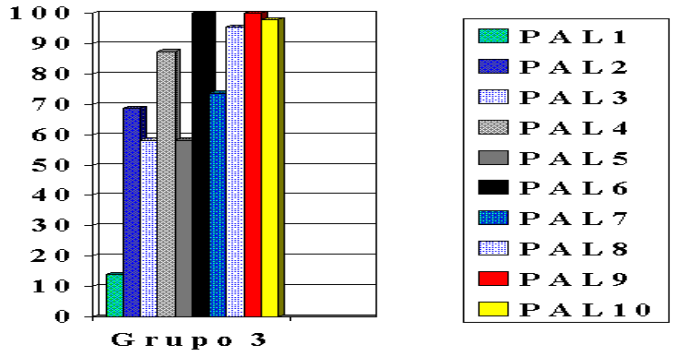


FIGURA 6.2: Taxa de Reconhecimento do HMM para o Grupo 3

6.1.2 RNA

Na Figura 6.3 tem-se o modelo da RNA.

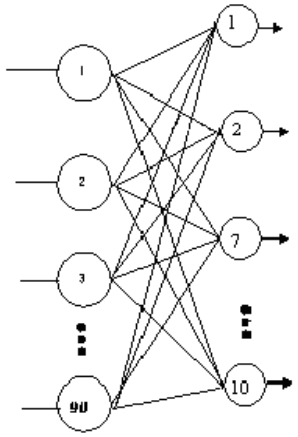


FIGURA 6.3 : Modelo de RNA Utilizado no Sistema Híbrido.

Na Tabela 6.4 encontra-se a matriz de confusão da Rede Neural para o Grupo 1 e na Tabela 6.5 para o Grupo 3.

TABELA 6.4: Matriz de Confusão da RNA para o Grupo 1

	P2	P7	P9	P10
P1		3	1	
P2				1
P3				
P4	1			

TABELA 6.5: Matriz de Confusão da RNA para o Grupo 3

	P2	P3	P4	P5	P6	P7	P8	P9	P10
P1	1		1		2	33		3	
P2		1		1	2	3			6
P3				1			4		5
P4				4		1	2		
P5									3
P7								5	
P8			2	1					
P10					1				

De acordo com os resultados da Tabelas 6.3 e 6.5 percebe-se que o desempenho dos modelos diminui com a mudança do ambiente de gravação. Nas Figuras 6.4 e 6.5 tem-se a taxa de reconhecimento por palavras.

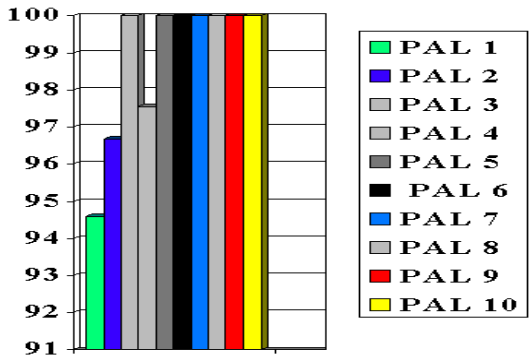


FIGURA 6.4: Taxa de Reconhecimento para o Grupo 1

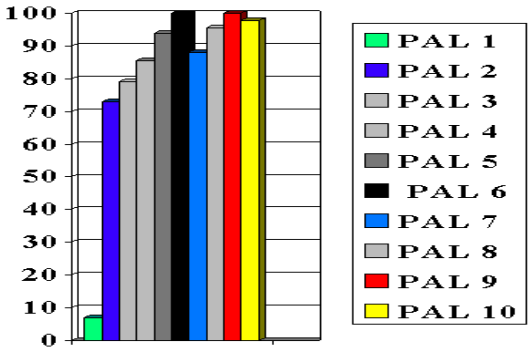


FIGURA 6.5: Taxa de Reconhecimento para o Grupo 3

Na Figura 6.6 tem-se a taxa de reconhecimento para o HMM e para a Rede Neural.

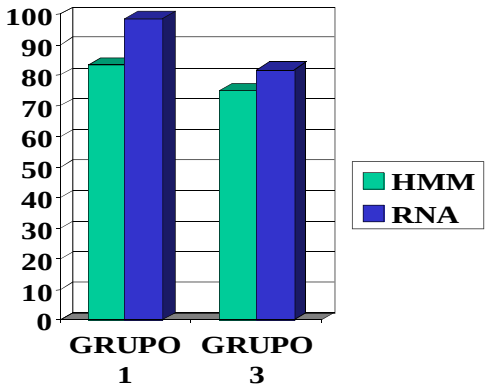


FIGURA 6.6: Comparação entre o HMM e a RNA

6.2 ESTRUTURAS DO SISTEMA HÍBRIDO

A estrutura híbrida pesquisada neste trabalho encontra-se na Figura 6.7.

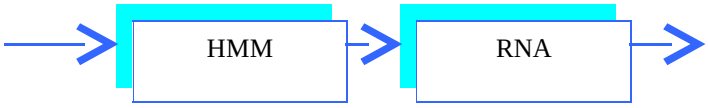


FIGURA 6.7: Modelo Híbrido

Para implementar o sistema híbrido foram analisados alguns parâmetros do HMM, quais sejam:

- Matriz de transição entre os estados.
- Matriz de probabilidades das observações.
- Verossimilhança por estado.
- Função de duração do estado e Verossimilhança normalizada.

6.2.1 Sistema Híbrido Utilizando a Matriz de Transição entre os Estados

A matriz de transição entre os estados contém informações temporais da palavra modelada. Para verificar a sua importância em um pós-processamento, as transições foram treinadas por uma rede neural.

A matriz de transição de um modelo com 6 estados, segundo o Modelo de Bakis, fica definida como:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 & 0 \\ 0 & a_{22} & a_{23} & a_{24} & 0 & 0 \\ 0 & 0 & a_{33} & a_{34} & a_{35} & 0 \\ 0 & 0 & 0 & a_{44} & a_{45} & a_{46} \\ 0 & 0 & 0 & 0 & a_{55} & a_{56} \\ 0 & 0 & 0 & 0 & 0 & a_{66} \end{bmatrix}$$

Para formar a matriz de entrada da Rede Neural, a matriz de transição de uma locução foi transformada em vetor com 15 linhas, da seguinte forma:

- Os elementos iguais a zero da matriz de transição, A, são iguais para todos os modelos com 6 estados, assim foram desconsiderados na montagem da matriz de entrada da Rede Neural.
- Os elementos diferentes de zero das primeira coluna foram postos por cima dos elementos diferentes de zero da segunda coluna e assim por diante; formando dessa forma a matriz de entrada da Rede Neural.

Na Figura 6.8 tem-se o primeiro sistema híbrido.

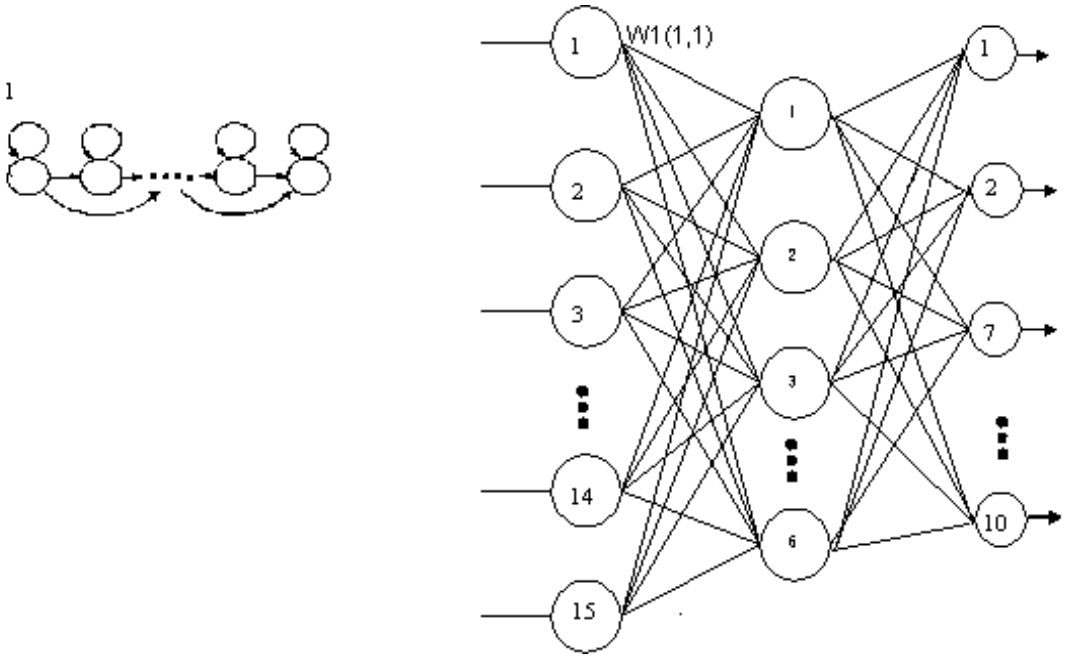


FIGURA 6.8: Sistema Híbrido com a informação temporal do HMM. A matriz de saída é formada pelo código de bits ortogonais da Tabela 5.8.

O treinamento da rede neural do sistema híbrido foi repetido 5 vezes, e em nenhuma das vezes a rede atingiu o erro desejado, desse modo pode-se concluir que:

- Os dados da matriz de transição devem sofrer um outro tratamento para facilitar o treinamento.
- ou, no pior caso, os dados da matriz de transição não são úteis para um sistema híbrido.

6.2.2 Sistema Híbrido Utilizando a Matriz de Probabilidades de Observações

A matriz de probabilidade de observações contém informações acústicas do modelo e diferentemente da matriz de transição, a Rede Neural convergiu nas 5 vezes.

A matriz de probabilidade de observações tem dimensões de 60× 6, isto é, a probabilidade de cada uma das 60 observações pertencerem a cada um dos 6 estados. Dessa forma, a matriz de entrada da Rede Neural contém vetores com 360 linhas.

De acordo com as Tabelas 6.2 e 6.3 existem 3 grupos de confusão e assim foi construído um Sistema Híbrido para cada um dos grupos de confusão.

Na Figura 6.9, tem-se o Sistema Híbrido para um grupo de confusão.

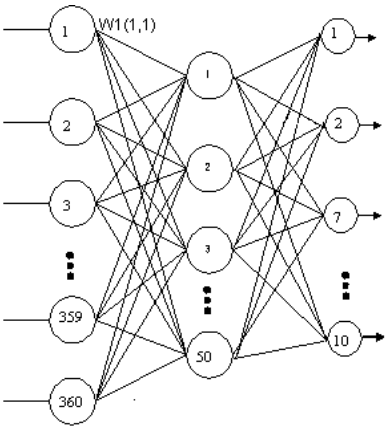


FIGURA 6.9: Sistema Híbrido com informações acústicas do HMM.

Na Figura 6.10, tem-se a taxa de reconhecimento para o Sistema Híbrido da Figura 6.9.

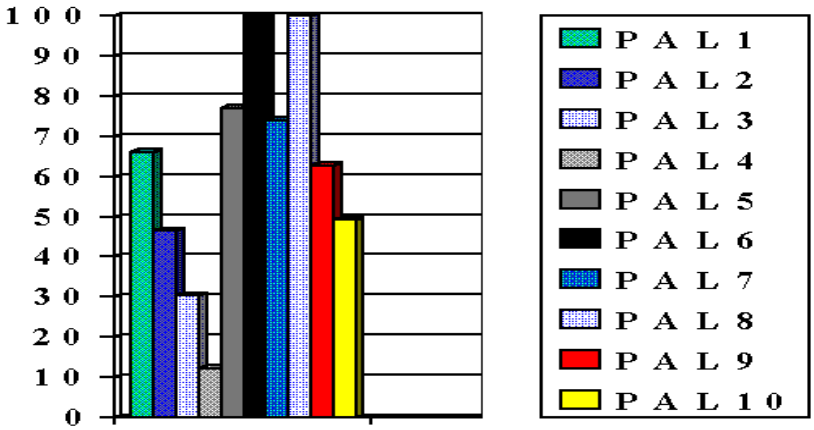


FIGURA 6.10: Taxa de Reconhecimento para o Segundo Sistema Híbrido

De acordo com a Figura 6.10, pode-se chegar as mesmas conclusões obtidas com a matriz de transição. Entretanto, todos os dados de treinamento foram corretamente classificados, daí pode-se dizer que: Para obter um desempenho satisfatório utilizando Redes Neurais devem ser considerados tanto a estrutura da Rede Neural quanto as *features* , que neste caso foram as distribuições acústicas.

6.2.3 Sistema Híbrido Utilizando a Verossimilhança por Estado

O algoritmo de *Viterbi* fornece dois parâmetros:

- a seqüência ótima de estados,
- a verossimilhança final.

Após a utilização do *Viterbi*, encontra-se a verossimilhança por estado combinando a sequencia ótima de estados com as matrizes de transição e de probabilidades.

O sistema híbrido implementado com esta característica não aumentou o desempenho do HMM.

6.3 SISTEMA HÍBRIDO UTILIZANDO A FUNÇÃO DE DURAÇÃO DE ESTADO E A VEROSSIMILHANÇA NORMALIZADA [34]

A duração de estado é utilizada no pós-processamento para aumentar a taxa de reconhecimento. Um método de introduzir a informação de duração de estado é medir diretamente das seqüências segmentadas pelo algoritmo de Viterbi. Dessa forma o procedimento para o reconhecimento torna-se o seguinte:

- Primeiro utiliza-se o algoritmo de Viterbi para se achar a melhor segmentação e verossimilhança associada;
- É feita uma contagem das observações que permanecem nos estados, e encontra-se o valor médio e a variância de cada palavra do vocabulário;
- Para uma representação paramétrica, a densidade de probabilidade da duração de estado é definida como:

$$p(d) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left[-\frac{(d-\mu)^2}{2\sigma^2} \right] \tag{6.1}$$

onde:

- d é a duração da palavra;
- μ é a média das durações das palavras;
- σ é o desvio padrão das palavras e
- p(d) é a probabilidade de duração da palavra, dado o modelo

A verossimilhança normalizada é definida como:

$$\log \hat{P}(q, O/\lambda) = \log P(q, O/\lambda) + \alpha_d \sum_{j=1}^{N} \log\left[p_j\left(d_j\right) \right] \tag{6.2}$$

onde α_d é um fator de peso, e d_j é a duração do estado j ao longo do caminho ótimo obtido pelo algoritmo de Viterbi. O custo computacional deste tipo de pós-processamento é desprezível e a experiência tem mostrado que seu desempenho no reconhecimento é tão bom quanto o obtido com a implementação da duração ao longo do processamento.

6.3.1 Sistema Híbrido Utilizando HMM's com um Número Variado de Estados

De acordo com a taxa de reconhecimento do HMM foi selecionado o melhor modelo para cada palavra. Após a seleção, foi construída a matriz de entrada da RNA do Sistema Híbrido da seguinte forma:

1. A locução passa através dos dez modelos, referentes às dez palavras, e utilizando as Equações 6.1 e 6.2 obtém-se as durações de estados e verossimilhanças normalizadas.
2. As durações e verossimilhanças foram definidas como: PD_{ij} e $PROB_{ij}$. Onde o índice i refere-se a locução e o índice j ao modelo.
3. Como existem 10 palavras a serem modeladas e 10 modelos tem-se 100 combinações possíveis. Por meio destas combinações o sistema passou a ter informações sobre o contexto formado pelas dez palavras.
4. Utilizando as combinações foram criados 100 arquivos com extensão *.mat* contendo informações sobre: a média e variância das observações, da verossimilhança e da duração do estado.

Veja a Figura 6.11:

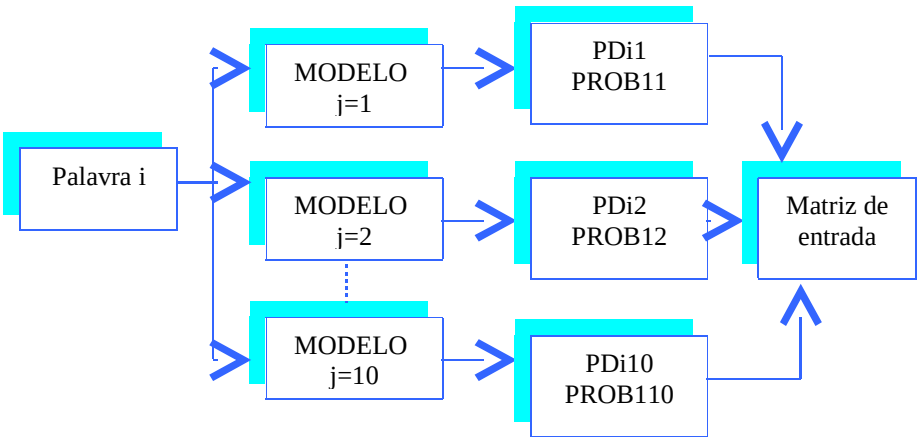


FIGURA 6.11: Montagem da Matriz de entrada

O modelo Híbrido pode ser visto abaixo:

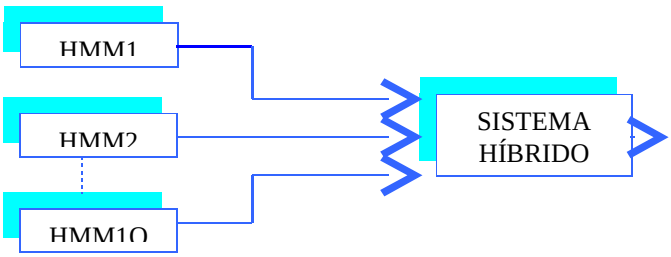


FIGURA 6.12: Modelo Híbrido utilizando HMM's com vários estados

O desempenho deste Sistema Híbrido foi inferior ao do HMM.

6.3.2 Sistema Híbrido Utilizando HMM's com o mesmo Número de Estados

Uma problema apresentado pelos Sistemas Híbridos foi a lentidão na fase de reconhecimento. Então para diminuir o tempo foi levado em consideração a tabela de confusão e foram treinados três híbridos.

Veja como foi feito o treinamento para o terceiro Grupo de Confusão:

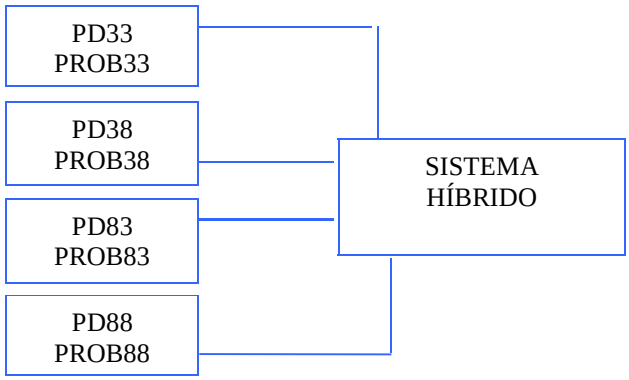
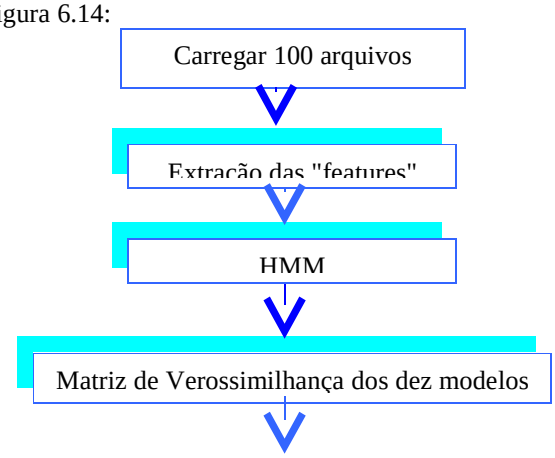


FIGURA 6.13: Treinamento do Sistema Híbrido para o grupo de Confusão 3.

Já o algoritmo de reconhecimento ficou definido como :

- Primeiramente carrega-se os 100 arquivos de dados para que se possa utilizar as variâncias e médias que farão parte dos cálculos da matriz de entrada da Rede Neural do Sistema Híbrido.
- Utilizando-se o algoritmo de Viterbi, o HMM reconhece a palavra **i** como sendo a mais provável.
- Dependendo da palavra que for reconhecida será montada a matriz de entrada para a Rede Neural de acordo as Equações 6.1 e 6.2.
- Os valores de i e j para montar a matriz de entrada serão iguais e dependerão do grupo de confusão a que pertence a palavra reconhecida pelo HMM. Assim, por exemplo, se o HMM reconhecer a palavra 1 como sendo a palavra mais provável, as variáveis i e j assumirão valores iguais a 1,6 e 7.
- O menor valor da matriz de erros da Rede Neural indicará a palavra reconhecida

Veja a Figura 6.14:



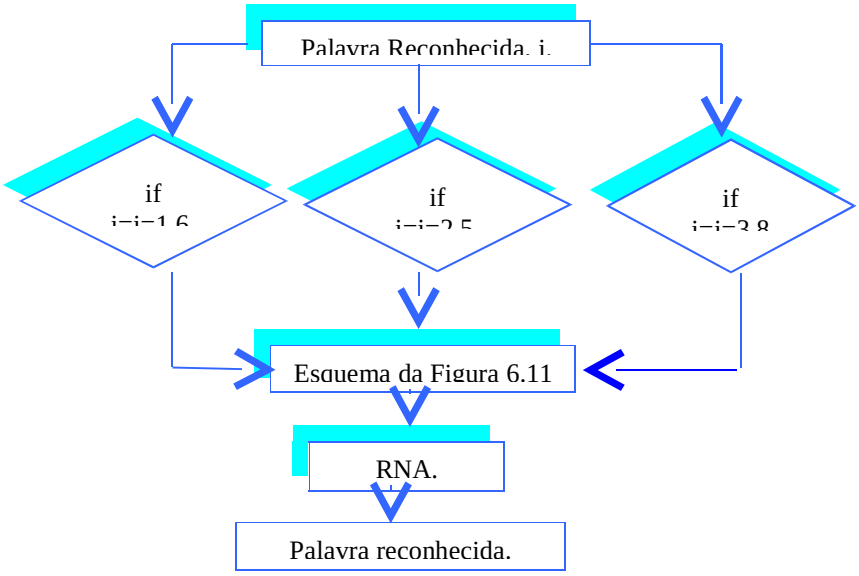


FIGURA 6.14: Modelo de Reconhecimento

Este sistema híbrido apresentou um desempenho inferior ao do HMM. Uma das razões foi a pouca informação contida na matriz de entrada da Rede Neural. Para melhorar o resultado foi proposto um sistema híbrido denominado HIB1

6.3.3 Sistema Híbrido - HIB1

Este sistema difere do anterior pelo algoritmo de reconhecimento onde o valor de j varia de 1 até 10 e a matriz de entrada passou a ter informações sobre as dez palavras. Este Sistema Híbrido, em muitas vezes, classificou erradamente as palavras em que o HMM corretamente classificava. Desse modo concluiu-se que a forma pela qual foi feito o reconhecimento não era a ideal e foi proposto um novo Sistema Híbrido, o HIB2.

6.3.4 Sistema Híbrido - HIB2 [33]

O algoritmo de reconhecimento deste Sistema Híbrido é baseado em intervalos de confiança, isto é,

$$P\left(\left|V-\mu\right| < 1{,}96\sigma_{\bar{x}}\right) = 0{,}95\,. \tag{6.3}$$

onde:

V= (a maior verossimilhança) - (a segunda maior verossimilhança)

μ = é a média sobre os valores de V obtidos com dados de treinamento.

σ= desvio padrão

$$\sigma_{\bar{x}}^{2} = \frac{\sigma^{2}}{n} \tag{6.4}$$

$$P = 1{,}96\sigma_{\bar{x}}^{2} \qquad (6.5)$$

Veja a Figura 6.15

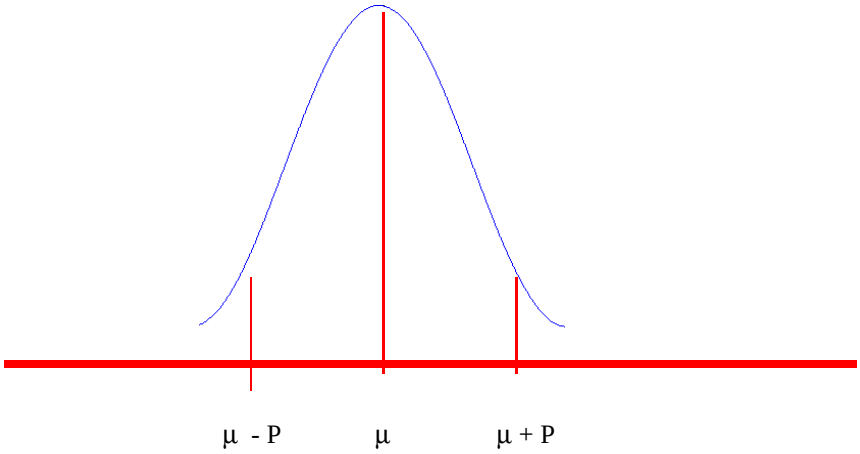
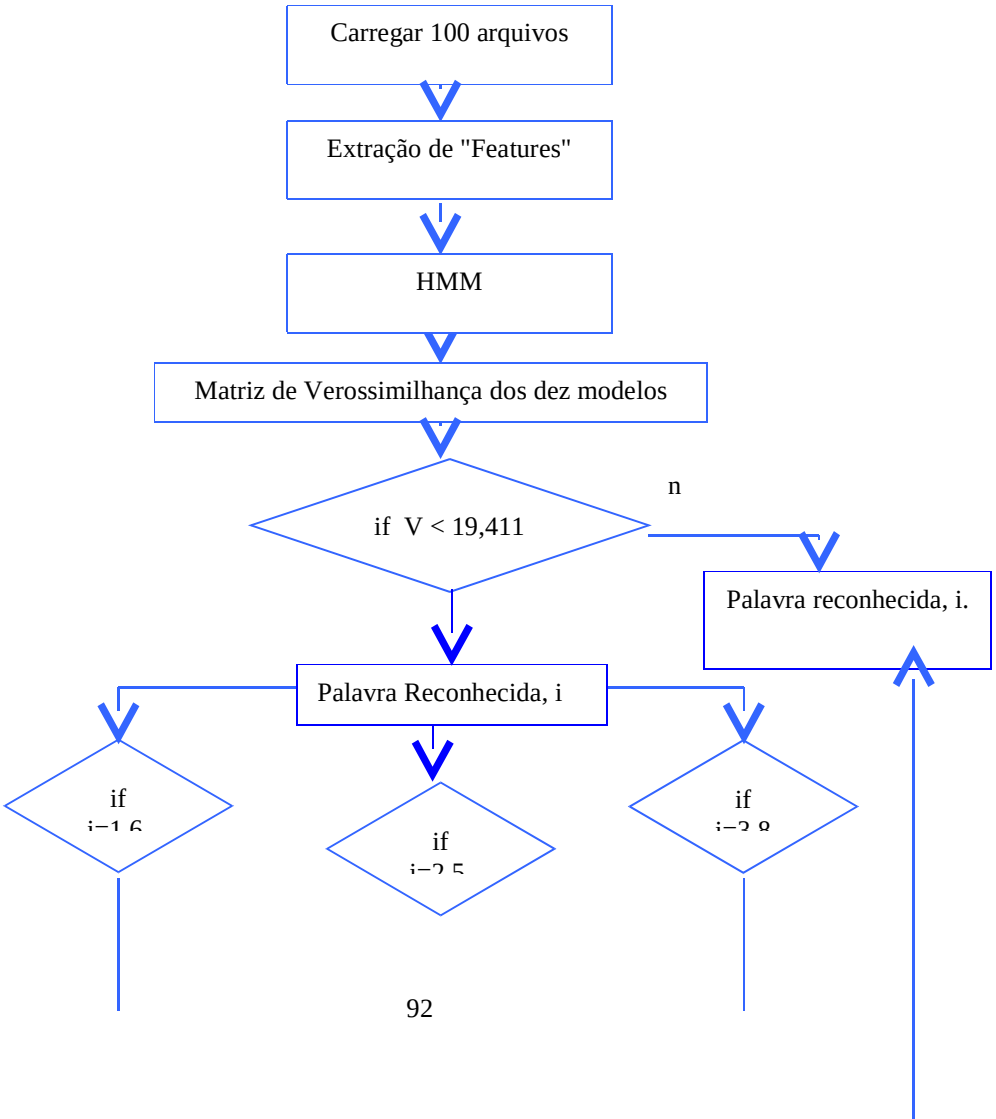


FIGURA 6.15: Intervalo de Confiança para o Sistema HIB2

De acordo com [illegible] a 19,411, assim se V for menor do que 19,411 a palavra reconhecida pelo HMM não é confiável e o Sistema Híbrido deverá ser utilizado para aumentar o grau de confiança da palavra reconhecida. Veja o algoritmo de reconhecimento mostrada na Figura 6.16.



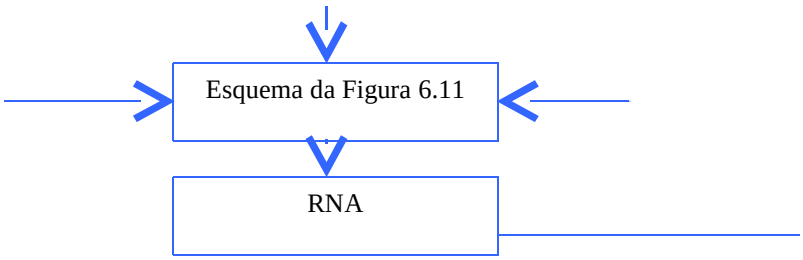


FIGURA 6.16: Modelo de Reconhecimento para o Sistema HIB2

Como este modelo de reconhecimento o Sistema Híbrido para o grupo 1 teve um desempenho superior ao do HMM, entretanto para o Grupo 3 foi observado que na matriz de confusão apareceram outros grupos de confusão. Para minimizar este problema foi proposto um novo Sistema Híbrido denominado HIB3.

6.3.5 Sistema Híbrido - HIB3

O algoritmo de reconhecimento para este Sistema Híbrido é mostrado na Figura 6.17

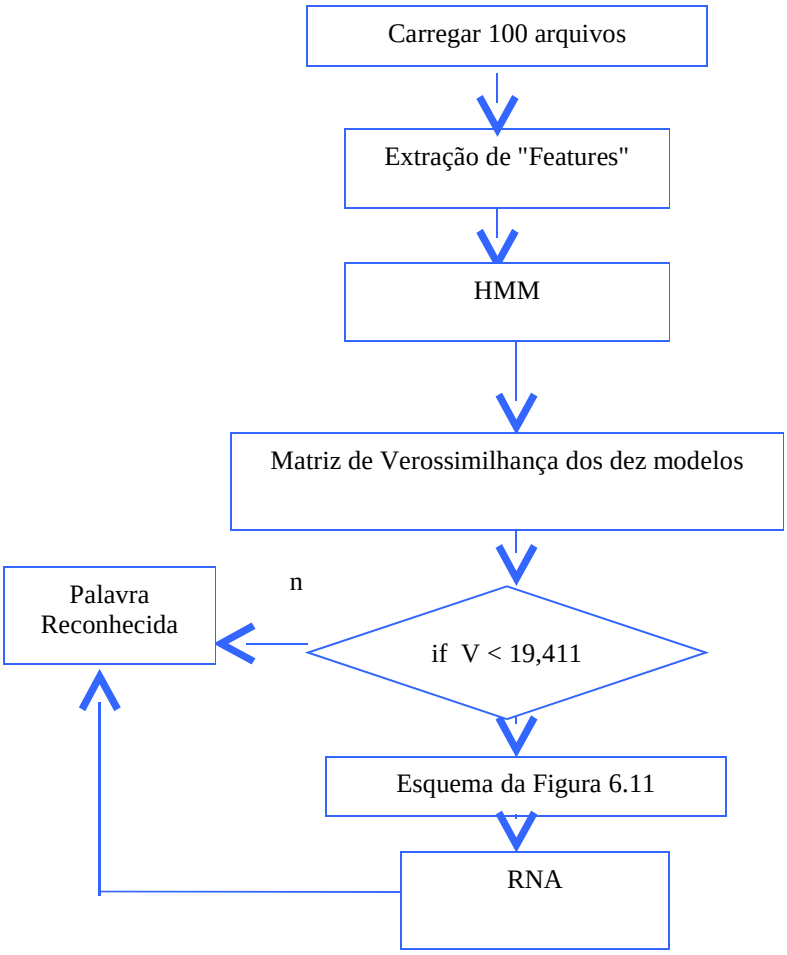


FIGURA 6.17: Modelo de Reconhecimento para o Sistema Híbrido - HIB3

O desempenho deste Sistema Híbrido foi superior aos demais, entretanto o tempo de reconhecimento foi elevado. Na Tabela 6.6 tem-se o tempo de reconhecimento para o HMM, a RNA e o HIB3.

TABELA 6.6: Tempo de Reconhecimento

	TEMPO
RNA	9,83 s
HMM	109 s
HIB3	1071s

Na Figura 6.18, a Tabela 6.6 pode ser vista graficamente

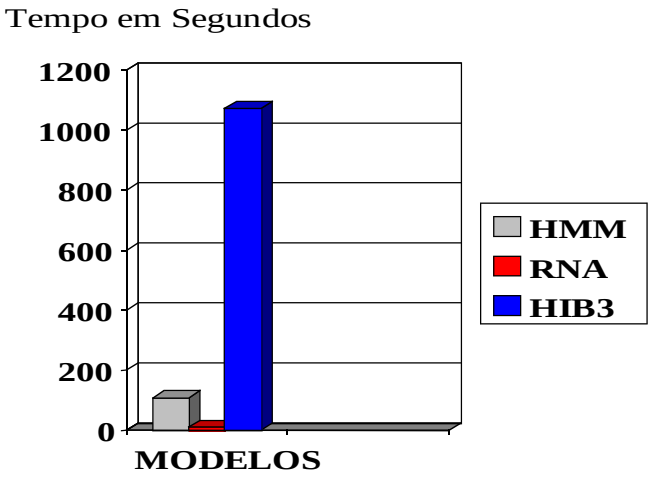


FIGURA 6.18: Tempo de Reconhecimento

Nas Tabelas 6.7 e 6.8 tem-se as matrizes de confusão para os Grupos 1 e 3 do Sistema Híbrido HIB3

TABELA 6.7: Matriz de Confusão do Modelo HIB3 para o Grupo 1

	P1	P2	P5	P6	P7	P8	P10
P1				3	9	1	1
P2							2
P3							1
P4							1
P5							3
P6							
P7	1						
P8							
P9							
P10		1	1				

TABELA 6.8: Matriz de Confusão para o Grupo 3

	P1	P4	P6	P7	P8	P9	P10
P1			1	4		1	
P2	2				1		18
P3							
P4							
P5							
P6							
P7	25					2	13
P8	1	4					3
P9							
P10	1						

Na Figura 6.19 tem-se a taxa de reconhecimento por palavra para os Grupos 1 e 3.

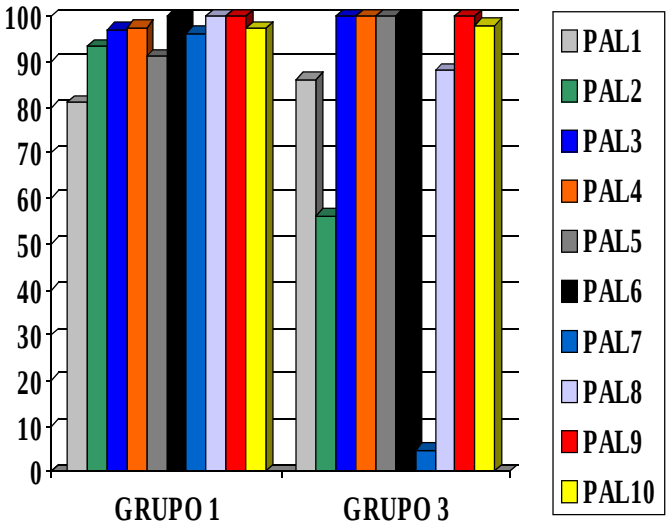


FIGURA 6.19: Taxa de Reconhecimento do HIB3 para o Grupo 1 e 3

Na Figura 6.20 tem-se as taxas de reconhecimento para o HMM, a RNA e o Sistema Híbrido.

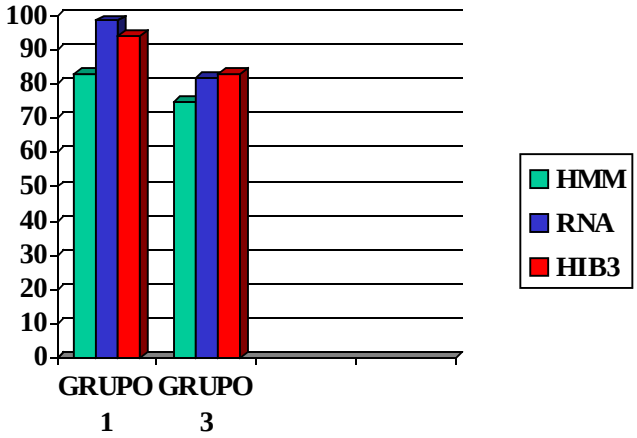


FIGURA 6.20: Comparação entre o HMM, a RNA e o HIB3

Como já foi dito, a escolha do HMM foi feita de tal forma que se pudesse ver a melhoria com a implementação de um Sistema Híbrido. Dessa forma a matriz de entrada da Rede Neural teve informações obtidas de um HMM com um desempenho insatisfatório. Por meio de um estudo sobre a montagem da matriz de entrada e de como se fazer o reconhecimento, esta situação foi revertida e o desempenho do Sistema Híbrido tornou-se melhor que o da RNA.

6.3.6 Sistema Híbrido - HIB4

O Sistema HIB4 utiliza uma Rede Neural denominada RNA1 para aumentar a informação contextual do modelo. Veja a Figura 6.21:

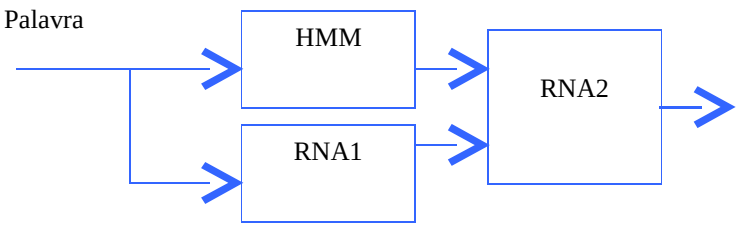


FIGURA 6.21: Modelo Híbrido HIB4

Na Tabela 6.9 tem-se a matriz de confusão do HIB4 para o Grupo 3

TABELA 6.9: Matriz de Confusão do Modelo HIB4 para o Grupo 3

	P1	P2	P4	P5	P7	P8	P9	P10
P1					34		2	
P2						1		2
P3		2			1	3		1
P4				2		1		
P5			2					
P6								
P7	3						1	
P8								
P9								
P10		1		1		1		

Na Figura 6.22 pode ser vista a taxa de reconhecimento por palavra

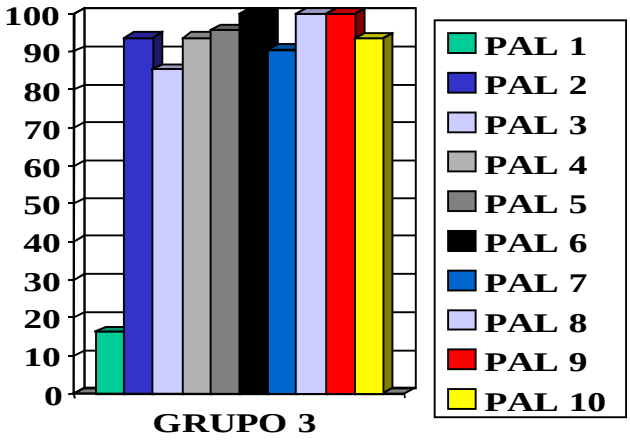


FIGURA 6.22: Taxa de Reconhecimento

6.3.7 Sistema Híbrido - HIB5

O modelo HIB5 é descrito na Figura 6.23. Assim como no HIB3 e HIB4 o modelo HIB5 apresentou problemas que foram solucionados com a normalização dos dados.

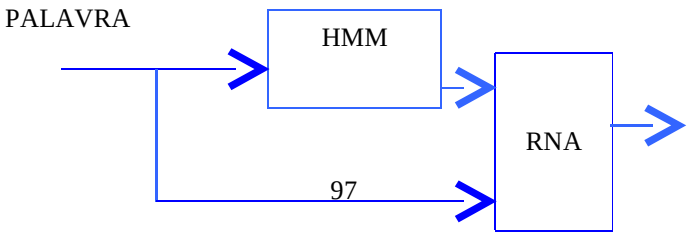


FIGURA 6.23: Modelo Hibrido HIB5

Na Tabela 6.10 encontra-se a matriz de confusão para o modelo HIB5

TABELA 6.10: Matriz de Confusão do Modelo HIB5 para o Grupo 3

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
P1						2	27		9	
P2			1					2		
P3		2		3				6		
P4			1		3			1		
P5			1			1				4
P6										
P7									15	
P8										
P9										
P10					1					

Na Figura 6.24 tem-se a taxa de reconhecimento para o Sistema HIB5.

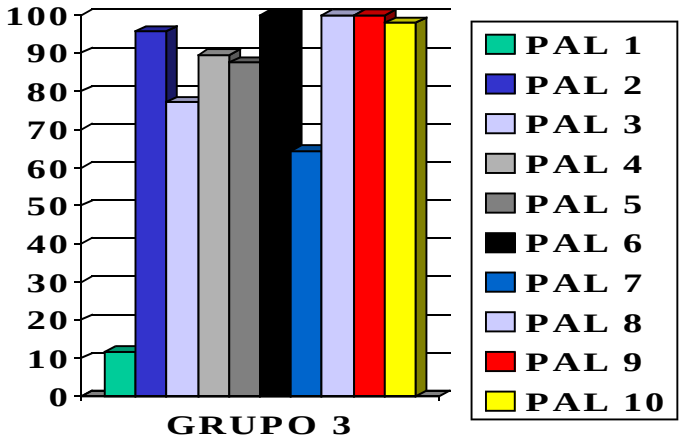


FIGURA 6.24: Taxa de Reconhecimento do HIB5 para o Grupo 3

6.4 OBJETIVO

De acordo com os resultados alcançados com a implementação do Sistema Híbrido pode-se concluir que os objetivos traçados no inicio do compêndio foram alcançados, isto é, o desempenho dos Sistemas Híbridos foi superior aos do HMM e da RNA.

Veja a Figura 6.25

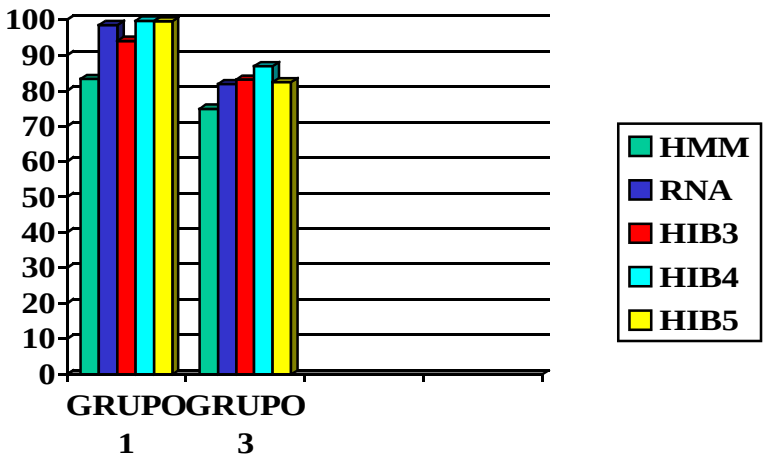


FIGURA 6.25: O desempenho do Sistema Híbrido foi superior ao do HMM e da Rede Neural

CAPÍTULO VII

CONCLUSÃO

Os sistemas de reconhecimento possuem inúmeras limitações que são agravadas pelo pouco conhecimento das características da voz humana. No desenvolvimento deste trabalho foram observados inúmeros problemas, tais como no:

- Pré-Processamento

- O algoritmo de *endpoint* não oferece bons resultados.
- As *features* utilizadas não são tão robustas com relação: a alteração do ambiente de gravação, estado emocional, sexo e entonação do locutor.

- Tamanho de cada janela e a superposição entra elas.
- Um maior banco de dados para melhor validação dos dados de teste

- HMM

- Aprofundar o estudo sobre o número de estados e de Gaussianas por estado.
- Que base utilizar no algoritmo de Viterbi do HMM.
- Qual o fator de convergência que diminui a relação custo/beneficio.
- Realizar um estudo mais aprofundado sobre a hipótese de independência que desconsidera qualquer informação de contexto e
- A hipótese de Markov de primeira ordem, que provoca uma modelagem inadequada das coarticulações.
- Aprofundar o estudo sobre a temporacidade do modelo.

- RNA

- Algoritmos de treinamento mais rápidos e mais robustos com relação a escolha dos parâmetros de treinamento.
- Aprofundar os estudos sobre as Redes Neurais do tipo *Time Delay* para introduzir uma boa modelagem temporal.
- Término do treinamento da Rede Neural.

Devido a existência destes problemas, a procura do modelo ideal foi substituída pela procura de um modelo com a menor estrutura possível que diminua o tempo de treinamento e mantenha o desempenho satisfatório. Entretanto com a infinidade de combinações entre os parâmetros, afirmar que este é o modelo ideal ou o de menor estrutura não é correto, então foi necessária a utilização de um Sistema Híbrido para aumentar a taxa de reconhecimento.

Nesses sistemas as Redes Neurais foram utilizadas para a modelagem acústica ou de duração de estado e o HMM para prover a modelagem temporal das locuções de voz.

A utilização dos Sistemas Híbridos aumentou a taxa de reconhecimento, entretanto também aumentou o tempo de reconhecimento mantendo a relação custo / benefício igual as do HMM e da RNA. Com uma possível otimização e implementação em C++ do algoritmo do Sistema Híbrido com certeza essa relação diminuirá.

Finalmente como sugestão para trabalhos futuros :

- Aprofundar os estudos sobre os problemas acima mencionados;
- Pesquisar outros modelos híbridos utilizando HMM e RNA;
- Pesquisar a utilização de outros modelos para o RAV, tais como a Lógica *Fuzzy*.