Uso de Técnicas Neurais para o Reconhecimento de Comandos de Voz

*Suelaine dos Santos Diniz**
*Antonio Carlos Gay Thomé***

No artigo anterior (parte II) abordou-se o uso de técnicas neurais para o Reconhecimento da Palavra Isolada – RPI: Neste artigo (parte III) são apresentados o sistema proposto para o RPI, o discriminante linear de Fisher usado para a seleção das características mais relevantes do sinal de voz, os modelos neurais concebidos para o estudo de caso realizado e os resultados alcançados.

PARTE III

USO DOS MODELOS NEURAIS NO RECONHECIMENTO DA PALAVRA ISOLADA

SISTEMA PROPOSTO

O sistema ora proposto, visa reconhecer um número finito de palavras isoladas de forma independente do locutor – pronunciadas somente por locutores do sexo masculino –, utilizando como classificador final três técnicas de redes neurais.

O vocabulário escolhido para estudo de caso e avaliação dos modelos, faz parte dos principais comandos presentes no mercado de aparelhos eletrônicos para controle de vídeo cassete. A expansão do vocabulário, incluindo palavras com sonorização próxima às existentes, teve como objetivo introduzir ambigüidade e tornar o reconhecimento mais difícil. Assim, com um total de 28 locutores, foi construída uma base de dados totalizando 1400 locu-

* MC / IME.
** Cel. R/1 – Ph.D. / Purdue

ções com palavras do conjunto: **liga, desliga, siga, pausa, pare, grave, avance, apague, ejete** e **volte.**

A implementação dos diferentes modelos estudados foi realizada com base em uma plataforma computacional constituída de um computador da linha PC, marca *"Gateway2000"* 486DX2-66, com 16 Mb de memória, uma placa de som da "Creative Labs Inc." modelo *"Sound Blaster 16"* e um microfone diretivo da marca *"Creative"*, conectado à entrada "mic in" da placa de som e todos os algoritmos utilizados em todas as fases do sistema foram desenvolvidos no ambiente MATLAB, com o apoio "das *"Tollboxes"* de Redes Neurais e de Processamento de Sinais.

O desenvolvimento do sistema proposto, foi realizado segundo as três seguintes etapas:

- Aquisição de dados - tomada de cada uma das 1400 locuções, e gravação do arquivo binário com extensão *".WAV"*. Tomou-se o cuidado de não realizar as séries de repetição de um mesmo locutor em uma única ocasião;
- Pré-processamento do sinal - constituído pela normalização do sinal amostrado, determinação dos pontos extremos da locução, determinação do tamanho da janela para cada locução de forma que se obtivesse sempre um total de 120 janelas com um mínimo de 76% de superposição entre elas, extração das características do sinal e análise e seleção das características mais relevantes para a construção dos modelos;
- Construção do modelo - formatação dos dados de entrada para as redes neurais, treinamento e validação das mesmas.

A figura 1 resume em um diagrama em blocos as atividades presentes na construção deste sistema para RPI.

AQUISIÇÃO DE DADOS

A aquisição dos dados e conversão A/D do sinal de voz é feita utilizando a placa *"Sound Blaster 16"*, no formato mono, visto ser o mais apropriado para gravação de voz, com taxa de amostragem de 11025 Hz e quantização de amostragem em 16 bits, que proporciona gravação de alta qualidade e possibilita a análise do sinal de voz com faixa dinâmica de 45 dB. As gravações dos sinais de voz foram feitas em uma sala sem qualquer tipo de isolamento acústico e as repetições serão capturadas em dias e horários diferentes, de forma a mostrar a variabilidade na pronúncia do locutor.

Para o treinamento e a validação (teste) dos modelos foram utilizadas respectivamente 20 e 11 locutores, sendo que dos 11 escolhidos para validação, foram solucionadas 3 para participarem também do treinamento, evidentemente que com locuções distintas. A idade média dos locutores participantes estava na casa dos 30 anos, e das 1400 locuções disponíveis, um total de 1110 locuções foram escolhidas para o estudo de caso (1000 para treinamento e 110 para teste).

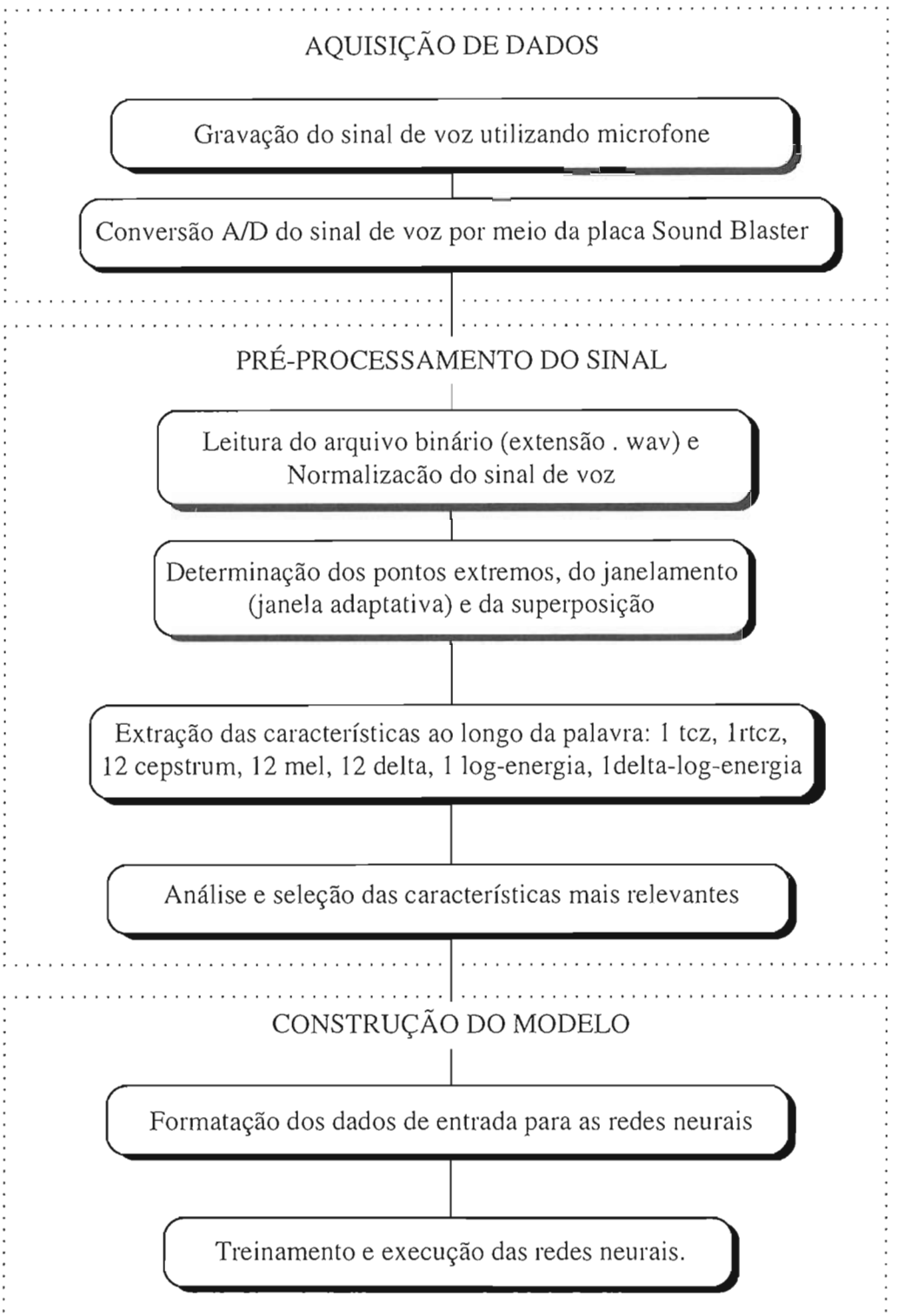

FIGURA 1 – Diagrama em blocos das atividades presentes no sistema RPI.

Os arquivos da base de voz criada podem ser identificados pela seguinte semântica: *l0xypzrq*, onde x e *y* representa o número do locutor, *z* o número da locução – palavra, onde: p0 = apague, p1 = liga, p2 = pare, p3 = grave, p4 = pausa, p5 = avance, p6 = siga, p7 = desliga, p8 = volte e p9 = ejete – e *q* o número da repetição desta palavra. Logo, como exemplo, pode-se citar o arquivo l 001p1r1.WAV, que significa: locutor 01, palavra 1 – liga – e primeira repetição desta palavra pronunciada por este locutor.

PREPARAÇÃO DOS DADOS

A fase de pré-processamento do sinal de voz consiste na leitura do arquivo binário e na normalização do sinal de voz amestrado. O arquivo com extensão "*.WAV.*" consiste de uma seqüência de 2 *bytes* – *16 bits* – por amostra, contendo um cabeçalho inicial de 44 *bytes*. Neste cabeçalho estão contidas informações tais como: freqüência de amostragem, modo de gravação – mono ou estéreo –, bits por amostra – 8 ou 16 níveis de quantização –, dentre outras.

O processo utilizado para extração do cabeçalho, utilizando o programa MATLAB é:

fid = fopen (arquivo. wav) % abre o arquivo; (1)

sinal = fread (fid, 'short') % leitura do arquivo ". *WAV"*, onde o parâmetro *short'* indica que esta leitura será feita de 2 em 2 *bytes;* (2)

freq = sinal (13) % contém a freqüência de amostragem 11025 Hz no caso deste trabalho; (3)

sinal = sinal (23: length (sinal) % eliminação do *"header "* de 44 *bytes.* (4)

Depois da obtenção do sinal amostrado é feita a normalização do mesmo entre os limites de amplitude de +/- 500 mV, ou seja 1 V de pico a pico. O objetivo desta normalização é equalizar as amplitudes dos sinais de voz gravados com níveis de volume diferentes. O processo utilizado nesta normalização, utilizando o programa MATLAB é:

$$L1 = max(sinal) \quad (5)$$

$$L2 = min(sinal) \quad (6)$$

$$topo = max(L1, abs(L2)) \quad (7)$$

$$sinal = sinal*.5 / topo \quad (8)$$

onde a divisão por topo faz com que o sinal lido fique com um máximo de 500 mV e um mínimo de -500 mV.

A partir deste ponto ainda restam três etapas para concluir o pré-processamento do sinal:

- deteção dos *"endpoints"* e eliminação de silêncios;

- ajustamento temporal;
- e extração das características acústicas.

DETERMINAÇÃO DOS *"ENDPOINTS"* E ELIMINAÇÃO DE SILÊNCIO

A grande vantagem da determinação dos pontos extremos *"endpoints" é* a redução total no tempo de processamento. Como exemplo, pode-se citar um sinal gravado da palavra **desliga** com 0.8169s de duração. Ao determinar o início e o fim deste sinal pelo algoritmo de Rabiner & Sambur – como pode ser visto na figura 2 –, o tempo de duração passou a ser de 0.55s, eliminando assim o período de silêncio ou o ruído de fundo que precedia e que sucedia a locução.

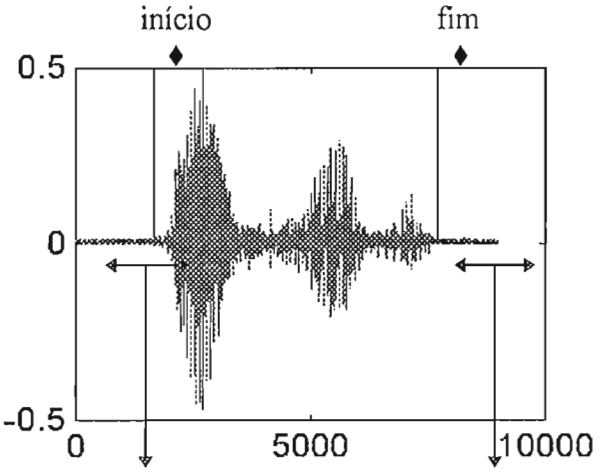

FIGURA 2 – Determinação do *"endpoint"* da palavra **desliga.**

O algoritmo de Rabiner & Sambur utilizado na localização desses pontos extremos baseia-se em duas medidas extraídas do sinal: a taxa de cruzamento de zero e a energia de tempo curto, obtidas em janelas de 10 ms de duração do sinal. Estas duas medidas são fáceis e rápidas de calcular, além de indicarem com precisão a presença ou a ausência da voz. Este algoritmo têm como objetivo principal estimar de forma rápida e eficiente o início e o fim de cada locução.

A figura 3 mostra a lógica do algoritmo de deteção de *"endpoint"*. Neste algoritmo é suposto que durante o intervalo de 100 ms iniciais e finais de gravação não há voz. Então, extraindo a média durante este intervalo, obtêm-se as medidas de ruído de fundo e de silêncio. Estas medidas incluem a medida e o desvio padrão da taxa da cruzamento de zero, e a medida da energia.

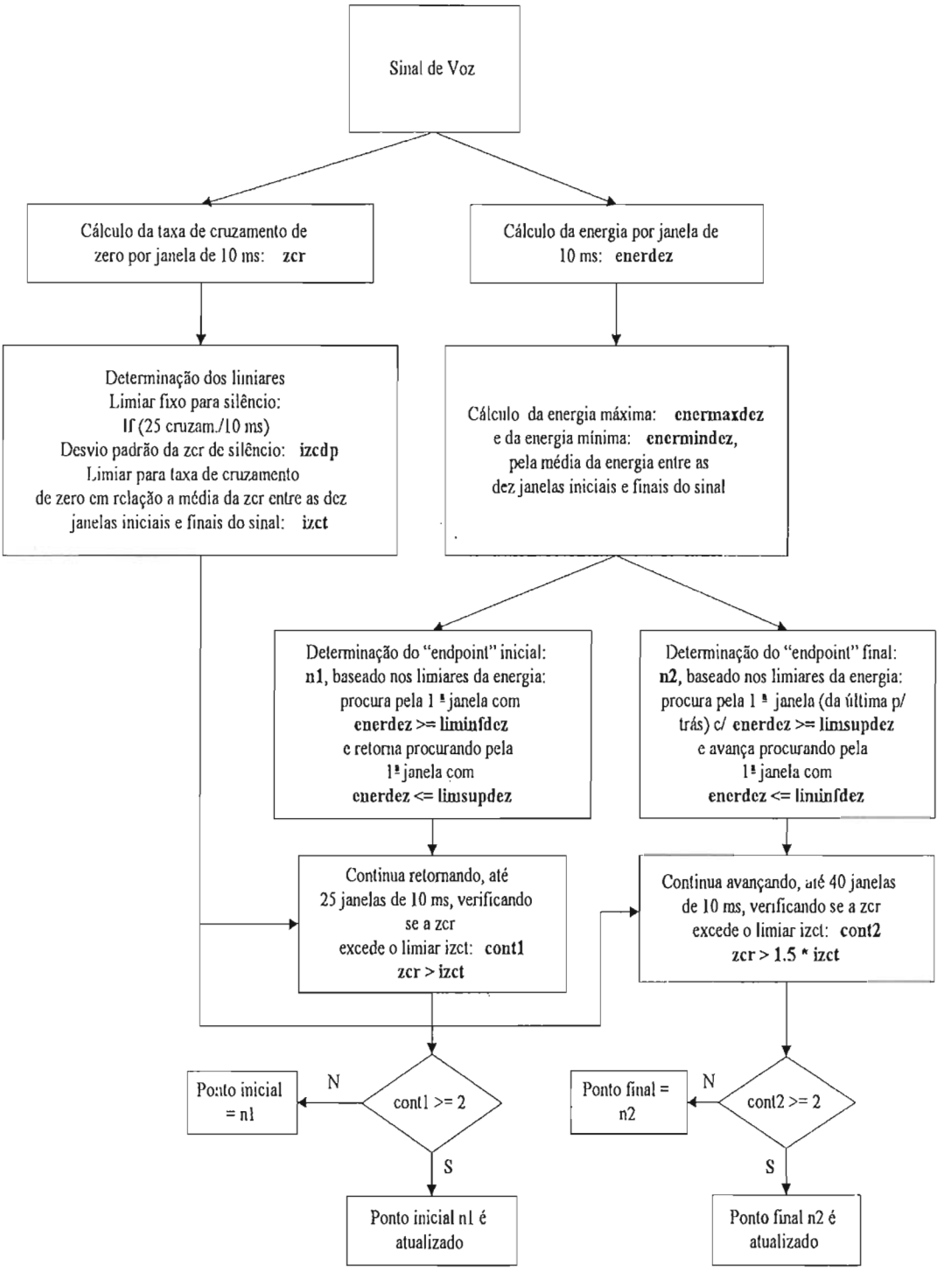

FIGURA 3 – Diagrama em bloco do algoritmo do *"endpoint"*.

O limiar da taxa de cruzamento de zero, *izct,* para sons não-vozeados foi escolhido como o mínimo de um limiar fixo, *lf* (25 cruzamentos de zero por 10 ms), e a soma da média da taxa de cruzamento de zero durante a zona assumida como de silêncio ou ruído de fundo *izcm,* mais duas vezes o desvio padrão da taxa de cruzamento de zero – neste período –, *izcdp, isto* é:

$$izct = min(lf, izcm + 2 * izcdp). \quad (9)$$

A estimativa dos valores da energia envolve o cálculo de dois limiares: limiar superior (limsupdez) e limiar inferior (liminfdez).

Estes limiares são obtidos em função das energias máxima (enermaxdez) e mínima (enermindez) e são imprescindíveis para o bom funcionamento do algoritmo. Os passos para determinação desses limiares são:

$$i1 = enemindez + 0.0\,1 * (enermardez - enermindez) \quad (10)$$

$$i2 = 2.2 * enermindez \quad (11)$$

$$\text{lim inf } dez = min(i1, i2) \quad (12)$$

$$\text{lim sup } dez = 2.2 * \text{lim inf } dez \quad (13)$$

A utilização do algoritmo de Rabiner & Sambur seguido de algumas modificações testadas em laboratórios, proporcionou uma melhor precisão na deteção dos pontos extremos da locução. Em vez de calcular as medidas de energia e taxa de cruzamento de zero somente nos 100 ms iniciais da locução, fez-se tal cálculo tornando por base a média dos 100 ms iniciais e finais desta. Com esta pequena alteração conseguiu-se melhorar bastante a deteção dos *"endpoints"* de locuções ruidosas, como a do comando **liga** mostrada na figura 4.

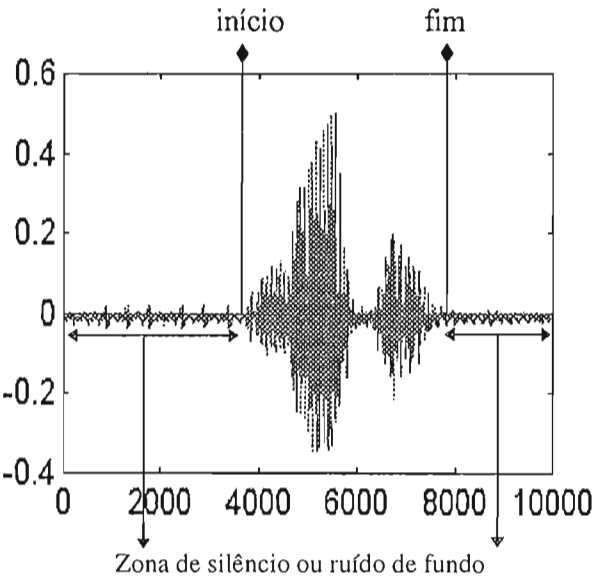

FIGURA 4 – Determinação do *"endpoint"* do comando **liga.**

Os sistemas RAV (Reconhecimento Automático de Voz) baseados em redes neurais do tipo *"feedfoward"* requerem um número de janelas sempre fixo e igual para toda e qualquer locução, independentemente do tempo de duração desta, obtendo-se assim vetores de mesmo comprimento para entrada da rede neural. Existem três métodos possíveis para este caso em questão:

- Decimação / Interpolação: consiste em comprimir (decimação) ou expandir (interpolação) o sinal de voz antes da extração das características, de tal forma que todas as locuções fiquem com o mesmo número de amostras. Este método apresenta a desvantagem de alterar a taxa de amostragem do sinal original; isto é devido à extração ou inclusão de amostras, e na conseqüente mudança do comprimento de cada locução.
- Superposição variável: consiste em fixar o tamanho de cada janela e determinar a superposição para cada locução, de tal forma que o número de janelas seja sempre o mesmo em todas as locuções, independentemente do tempo de duração desta. Neste método, as variações bruscas entre características extraídas de janelas adjacentes são evitadas devido ao aumento da correlação entre janelas sucessivas. Este método ainda apresenta a vantagem de não alterar o número de amostras do sinal original.
- Janela adaptativa: consiste em fixar a superposição e variar o tamanho de cada janela, de tal forma que o número de janelas de cada palavra seja o mesmo em todas as locuções, independentemente do tempo de duração desta. Esta técnica além das vantagens inerentes à superposição entre janelas consecutivas, proporciona um ajustamento temporal das locuções, ou seja: considerando-se que o alongamento no tempo ocorra de maneira uniforme, as janelas tenderão a corresponder aos mesmos fonemas pronunciados em frases idênticas com tempo de duração diferentes[1], isso

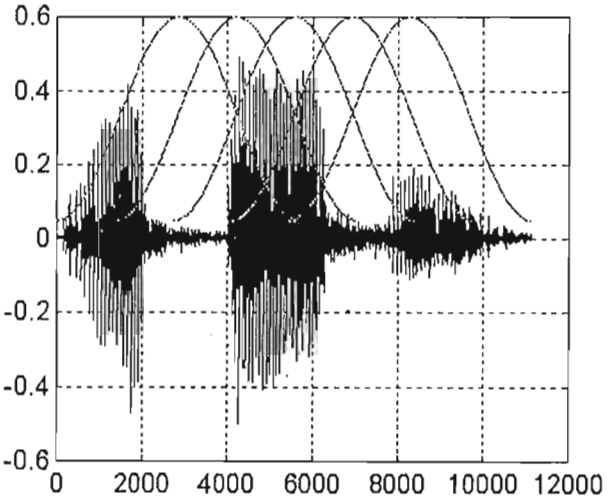

FIGURA 5 – Obtenção de 5 janelas adaptativas para o comando **apague** com 1.01 segundos de duração, 76% de superposição e 5681 amostras.

pode ser visto comparando-se a figura 5 onde é mostrada a forma de onda do comando **apague** com 1.01 segundos de duração, com 5 janelas de 0.5153s (5681 amostras) e superposição de 76%, com a figura 6 onde é mostrada a forma de onda de outra repetição do comando **apague** do mesmo locutor com 1.16 segundos de duração, com 5 janelas de 0.5918s (6525 amostras) e os mesmos 76% de superposição na janela de Hamming. Com base nesta comparação pode-se afirmar que as locuções maiores pronunciadas de forma mais lenta - serão avaliadas por janelas maiores, enquanto que locuções menores pronunciadas de forma mais rápida - serão avaliadas por janelas menores. Convém lembrar aqui que o sinal está sendo estudado ao longo de 120 janelas temporais; o uso das 5 janelas é só para efeito ilustrativo.

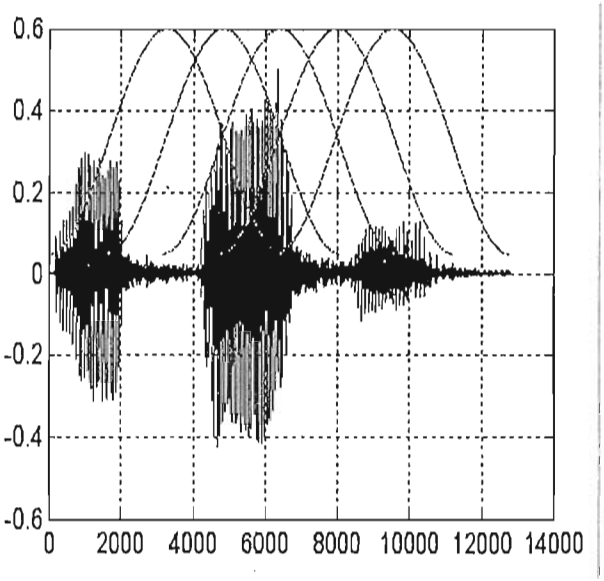

FIGURA 6 – Obtenção de 5 janelas adaptativas para o comando **apague** com 1.16 segundos de duração, 76% de superposição e 6525 amostras.

O número de amostras, ***na***, contido em uma janela – comprimento da janela – é calculado da seguinte maneira:

$$na = \frac{N}{(janela - 1)(1 - sup/100) + 1} \qquad (10)$$

onde: N é o número de amostras de uma determinada locução, ***janela*** é o número de janelas desejado e sup é o valor desejado da superposição, expressa em porcentagem.

SELEÇÃO DAS CARACTERÍSTICAS MAIS RELEVANTES PARA O PROBLEMA DO RECONHECIMENTO DA PALAVRA ISOLADA

CONSIDERAÇÕES

A análise completa e precisa do sinal de voz é de fundamental importância para o RAV. O resultado desta análise depende da representação adequada do sinal de voz – representação digital –, assim como da utilização das características fundamentais que representem com precisão o sinal em questão.

Em busca de um melhor desempenho do sistema, realizou-se uma análise estatística dos dados através do discriminante linear de Fisher[2,3] com o objetivo de identificar e escolher um subconjunto com as características mais relevantes e eficientes para o reconhecimento dos diferentes comandos.

ESTUDO COMPARATIVO DAS CARACTERÍSTICAS EXTRAÍDAS DO SINAL DE VOZ

Um dos problemas encontrados com maior freqüência na aplicação de técnicas estatísticas para reconhecimento de padrão é a dimensionalidade. Procedimentos que são analiticamente ou computacionalmente tratáveis num espaço de duas dimensões podem se tomar completamente impraticáveis em um espaço dimensional maior. Portanto, várias técnicas estatísticas têm sido desenvolvidas com vistas a possibilitar a redução da dimensão do espaço de *"features"*, na esperança de conseguir um problema mais tratável.

Pode-se reduzir a dimensionalidade de n dimensões a uma dimensão bem menor através da projeção das amostras; esta é a análise clássica do discriminante de Fisher[2,4,5].

O discriminante linear de Fisher – "J" - é definido segundo a expressão[32]:

$$J(w) = \frac{w^t S_B w}{w^t S_W w} \tag{15}$$

onde w é a matriz de transformação das amostras, S_B o espalhamento entre-classes e S_W o espalhamento intra-classes. As matrizes de espalhamento entre-classes e intra-classes, que conduzem à formulação final do discriminante de Fisher são respectivamente:

$$S_B = \sum_{i=1}^{n-1} \sum_{j=2}^{n} (m_i - m_j)(m_i - m_j)^t \tag{16}$$

&

$$S_W = \sum_i \tilde{S}_i^2 \quad \text{onde} \quad \tilde{S}_i^2 = \sum_{x \in X_i} (x - m_i)(x - m_i)^t \tag{16}$$

onde m é a média do conjunto de vetores da amostra.

Como o emprego do uso do discriminante de Fisher neste caso tem por objetivo apenas o de verificar a eficácia relativa de uma característica em relação a outra para o reconhecimento dos 10 comandos, a matriz de transformação w foi suposta como sendo uma matriz identidade, e assim a expressão do discriminante de Fisher reduziu-se a:

$$J = \frac{S_B}{S_w} = \frac{\text{variância entre - classes}}{\text{variância intra - classes}} \tag{17}$$

onde, as classes são caracterizadas pelos diferentes comandos que compõem o vocabulário utilizado. Assim, o numerador será maior quando os valores de uma determinada *'"feature"* estiverem mais dispersas, dentro do universo de locuções consideradas, e o denominador será menor quando os valores da *'feature"*, para várias repetições de um mesmo comando dita pelos diferentes locutores, possuírem poucas variações.

Cabe ressaltar que: quanto maior for a distância entre os centros de massa das classes, maior será a separação dos elementos projetados, ou seja, é desejado que a distância entre-classes seja alta e distância intra-classes seja baixa. O sentido prático dessa medida é possibilitar a geração de uma escala onde os maiores índices representam maior poder de discriminar uma classe da outra.

Foram analisadas neste caso 40 *"features"*: 12 coeficientes *"cepstrum"*, 12 *"mel-cepstrum"*, 12 *"delta-cepstrum"*, 1 *"delta-energia"*, 1 *"delta-log-energia"*, *1* taxa de cruzamento de zero normalizada e 1 relação da taxa de cruzamento de zero. As *"features"* foram extraídas em cada uma das 120 janelas utilizadas, com exceção da relação da taxa de cruzamento de zero que foi extraída ao longo de uma única janela contendo todas as amostras do sinal.

O valor de *"J"* foi então calculado para cada uma das 40 características isoladamente e selecionado um subconjunto com os maiores valores de *"J "*. A tabela 1 apresenta os resultados em ordem decrescente de *"J"* das *40 "features"* aplicadas sobre o conjunto de vetores usados para treinamento da rede neural.

Como visto na tabela 1, as 5 *melhores "features"* ordenadas com base no discriminante de Fisher para o conjunto dos 10 comandos utilizados neste sistema foram: 2º *"mel-cepstrum"*, 1º *"cepstrum"*, 3º *"mel-cepstrum"*, 1º *"mel-cepstrum"* e a relação da taxa de cruzamento de zero. Estas foram assim, as *"features"* escolhidas para formar o vetor representativo de cada uma das 1110 locuções utilizadas para o treinamento e teste das redes neurais. O comprimento total de cada padrão de entrada – observação – ficou em 481 e com o seguinte *"layout"* (figura 7):

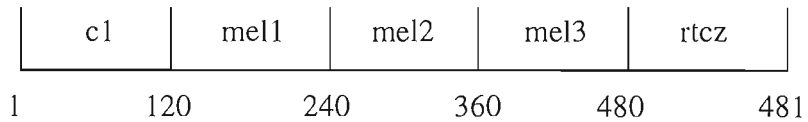

FIGURA 7 – Layout do comprimento total de cada padrão de entrada - característica.

TABELA 1: Resultado em ordem crescente da aplicação do discriminante de Fisher para 40 *'features"*.

CARACTERÍSTICAS	DISCRIMINANTE DE FISHER	
2º Mel-Cepstrum	1.6057	**1o**
1º Cepstrum	1.0696	**2o**
3º Mel-Cepstrum	0.9284	**3o**
1º Mel-Cepstrum	0.7393	**4o**
Relação da Taxa de Cruzamento de Zero	0.7271	**5o**
7º Mel-Cepstrum	0.6237	
4º Mel-Cepstrum	0.6139	
5º Cepstrum	0.5763	
6º Cepstrum	0.5698	
2º Cepstrum	0.3644	
6º Mel-Cepstrum	0.3259	
4º Cepstrum	0.3155	
9º Mel-Cepstrum	0.3006	
Taxa de Cruzamento de Zero - normaliz.	0.2843	
5º Mel-Cepstrum	0.2795	
3º Cepstrum	0.2711	
7º Cepstrum	0.1889	
11º Mel-Cepstrum	0.1851	
11º Cepstrum	0.1832	
10º Mel-Cepstrum	0.1601	
8º Mel-Cepstrum	0.1498	
9º Cepstrum	0.1394	
2º Delta-Cepstrum	0.1393	
Delta-Energia	0.1340	
12º Mel-Cepstrum	0.1217	
8º Cepstrum	0.1186	
10º Cepstrum	0.1168	
12º Cepstrum	0.0872	
3º Delta-Cepstrum	0.0814	
1º Delta-Cepstrum	0.0763	
4º Delta-Cepstrum	0.0740	
7º Delta-Cepstrum	0.0728	
6º Delta-Cepstrum	0.0453	
8º Delta-Cepstrum	0.0422	
9º Delta-Cepstrum	0.0416	
11º Delta-Cepstrum	0.0355	
5º Delta-Cepstrum	0.0334	
10º Delta-Cepstrum	0.0315	
12º Delta-Cepstrum	0.0200	
Delta-Log-Energia	0.0195	

onde: cl = 1º *"cepstrum"* (1 à 120); mel1 = 1º *"mel-cepstrum"* (121 à 240); mel2 = 2º *"mel-cepstrum"* (241 à 360); mel3 = 3º *"mel-cepstrum"* (361 à 480); rtcz .= relação da taxa de cruzamento de zero (481).

Para execução do programa "funcaof2.m", que permite o cálculo do discriminante de Fisher para cada *"feature"* – isoladamente –, é necessário que a *"feature"* em questão seja fornecida no seguinte formato; por exemplo:

feature = 1000 x 120

→ 1 "feature" * 120 janelas

→ 10 comandos * 5 repetições * 20 locutores

A lógica do programa "funcaof2.m" é mostrada no diagrama em blocos da figura 8.

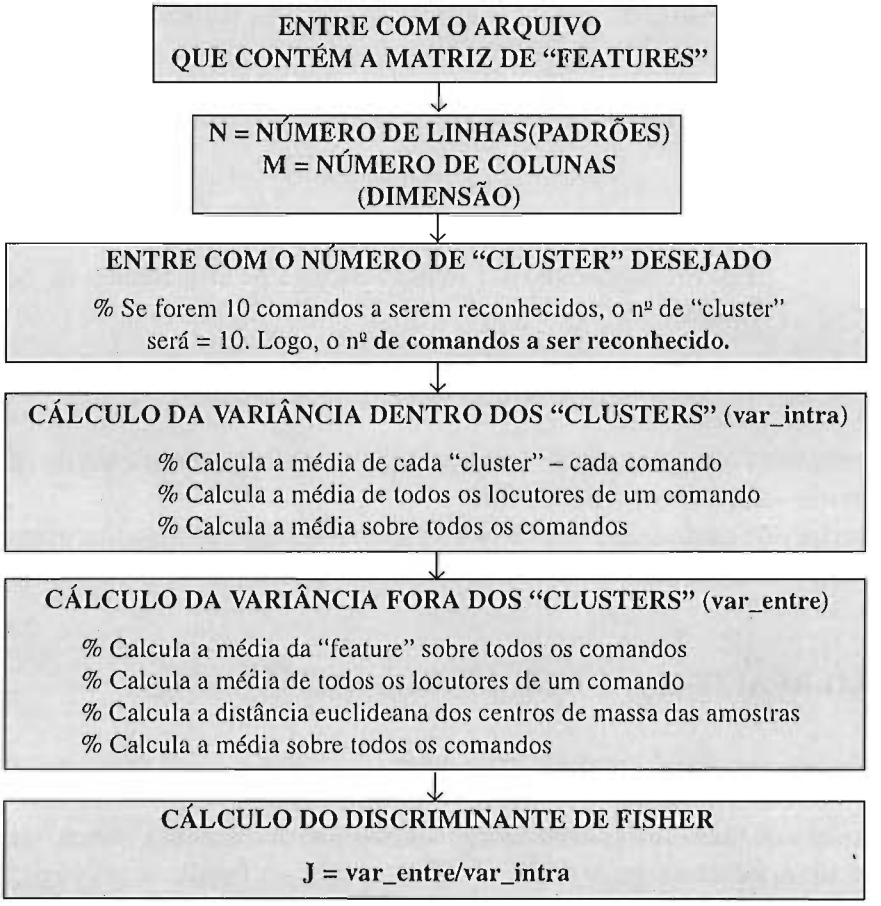

FIGURA 8 – Diagrama em blocos da implementação do programa funcaof2.m.

CONSTRUÇÃO DOS MODELOS

Depois de extraídas as principais características do sinal de voz, têm-se que montar as matrizes de treinamento e teste para entrada da rede neural. A matriz de treinamento foi gerada da seguinte forma:

1000 x 481

→ 1º *"cepstrum"* x 120 (nº de janelas) + 3 *"mel-cepstrum"* – 1º, 2º e 3º – x 120 (nº de janelas) + 1 relação da taxa de cruzamento de zero x 1 (nº de janelas)

→ 20 (locutores) x 10 (comandos) x 5 (repetições)

conforme mostra a figura 9.

Enquanto que a matriz de testes foi criada da seguinte forma:

110 x 481

→ 1º *"cepstrum"* x 120 (nº de janelas) + 3 *"mel-cepstrum"* – 1º, 2º e 3º – x 120 (nº de janelas) + 1 relação da taxa de cruzamento de zero x 1 (nº de janelas)

→ 11 (locutores) x 10 (comandos) x 1 (repetição)

onde as três primeiras locuções são de locutores que também participaram da fase de treinamento do sistema – dependentes do locutor.

As matrizes de saída usadas nas RNA's foram definidas de maneira a formar um conjunto ortogonal para os 10 comandos utilizados, com valores "0" e "1", conforme a tabela 2.

SIMULAÇÃO REALIZADA E RESULTADOS ALCANÇADOS

Realizou-se um estudo comparativo entre os modelos MLP simples com algoritmo *"backpropagation"*, *"radial basis"* e *"pshnn"* para o sistema RPI em questão.

No modelo de rede *"backpropagation"* utilizou-se treinamento "batch", taxa de aprendizagem adaptativa, momento igual a 0.75 – valor empírico – e função de propagação *"logística sigmoidal"* nas 3 camadas utilizadas. A configuração usada foi: 481 x 69 x 20 x 10, ou seja 481 características, 69 neurônios na 1ª camada escondida, 20 na 2ª camada escondida e 10 na

		NÚMERO DE CARACTERÍSTICA				
		1ª caract.	2ª caract.	3ª caract.	4ª caract.	5ª caract.
		Nº de janelas = 120				Janela = 1
LOCUTOR 1	**palavra 1 repetição 1**	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 1 repetição 2	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 1 repetição 3	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 1 repetição 4	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 1 repetição 5	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
⋮			⋮			
LOCUTOR 20	**palavra 1 repetição 1**	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 1 repetição 2	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 1 repetição 3	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 1 repetição 4	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 1 repetição 5	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
LOCUTOR 1	**palavra 2 repetição 1**	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 2 repetição 2	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 2 repetição 3	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 2 repetição 4	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 2 repetição 5	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
⋮			⋮			
LOCUTOR 20	**palavra 10 repetição 1**	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 10 repetição2	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 10 repetição3	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 10 repetição4	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz
	palavra 10 repetição5	C1 C1 ... C1	Mel1 .. Mel1	Mel2 .. Mel2	Mel3 .. Mel3	Rtcz

FIGURA 9 – Montagem da matriz de treinamento para entrada da rede neural do sistema em questão.

TABELA 2: Representação da saída desejada.

	CÓDIGOS ORTOGONAIS DE 10 BITS									
PALAVRAS	1	2	3	4	5	6	7	8	9	10
Liga	1	0	0	0	0	0	0	0	0	0
pare	0	1	0	0	0	0	0	0	0	0
grave	0	0	1	0	0	0	0	0	0	0
pausa	0	0	0	1	0	0	0	0	0	0
avance	0	0	0	0	1	0	0	0	0	0
siga	0	0	0	0	0	1	0	0	0	0
volte	0	0	0	0	0	0	1	0	0	0
ejete	0	0	0	0	0	0	0	1	0	0
desliga	0	0	0	0	0	0	0	0	1	0
apague	0	0	0	0	0	0	0	0	0	1

camada de saída. Para treinamento utilizou-se 1000 padrões de entrada, 100 para cada um dos 10 comandos considerados, enquanto que para teste utilizou-se 110 padrões de entrada, 11 para cada comando, sendo que três de locutores que também participaram do treinamento.

O *modelo "backpropagation"* convergiu com 36710 *"epochs"*, alcançando erro de 2,7662 durante o treinamento. Na tabela 3 observa-se o resultado obtido no teste – em porcentagem –, onde os comandos **pausa, avance, siga, volte, ejete** e **apague** apresentaram taxa de acerto de 100% – 11 locuções –, enquanto que os comandos **liga, pare, grave** e **desliga** apresentaram 81.8% – 9 locuções –, 90.9% – 10 locuções –, 90.9% e 90.9% de acerto, respectivamente. Na tabela 4 são apresentados os resultados do teste - em número de locuções reconhecidas para o comando testado - que obtiveram taxa de acerto menor que 100%, onde o 1º resultado é para o modelo *"backpropagation"*. Como exemplo, pode-se ver na tabela 4 que das 11 locuções testadas do comando **desliga**, 10 foram reconhecidas corretamente, enquanto que 1 foi reconhecida como **liga**.

No modelo *"radial basis"* utilizou-se uma camada com 10 gaussianas, a qual reduziu a dimensão inicial de 480 – antes do particionamento dos dados – para 40, dois estágios de redes do tipo MLP com treinamento *"batch"* e *algoritmo "backpropagation"*. Cada uma das

4 redes (uma rede para cada característica) do 1º estágio foi configurada com 10 x 40 x 20 x 10, ou seja, 10 características – dimensão – oriundas das respostas das funções gaussianas, 40 neurônios na 1ª camada escondida, 20 na 2ª camada escondida e 10 na camada de saída. O melhor resultado obtido foi com um número de classes igual ao número de comandos (10) do sistema. A rede do 2º estágio foi configurada com 40 x 55 x 20 x 10, ou seja, 40 características – dimensão – oriundas de todas as saídas obtidas durante o treinamento do 1º estágio da rede MLP. O 1º estágio do modelo apresentou erro [116 210 112 185] para 5000 *"epochs"*, erro este referente às seguintes características: 1º *"cepstrum"*, 1º, 2º e 3º *"mel-cepstrum"* respectivamente. Observou-se que as características mais relevantes para o reconhecimento dos 10 comandos foram em ordem crescente: 2º *"mel-cepstrum"*, 1º *"cepstrum"*, 3º *"mel-cepstrum"* e 1º *"mel-cepstrum"*, apresentando resultado semelhante ao obtido com o discriminante linear de Fisher.

Os padrões utilizados para treinamento e teste do modelo *"radial basis"* foram os mesmos do modelo *"backpropagation"* já descrito. Na tabela 3 observa-se o resultado obtido no teste – em porcentagem –, onde os comandos **liga, pausa, avance, siga, volte, ejete, desliga** e **apague** apresentaram taxa de acerto de 100% – 11 locuções –, enquanto que os comandos **pare e grave** apresentaram 90.9% – 10 locuções – de acerto. Na tabela 4 são apresentados os resultados do teste – em número de locuções reconhecida para o comando testado – que obtiveram taxa de acerto menor que 100%, onde o 2º resultado é para o modelo *"radial basis"*. Como exemplo, pode-se ver na tabela 4 que das 11 locuções testadas do comando **pare**, 10 foram reconhecidas corretamente enquanto que 1 foi reconhecida como **grave.**

TABELA 3: Porcentagem de acertos obtidos pelos modelos: *"Backpropagation", "Radial Basis" e "PSHNN" para* o reconhecimento dos 10 comandos.

COMANDOS	BACKPROPAGATION	RADIAL BASIS	PSHNN
LIGA	81.8%	100%	90.9%
PARE	90.9%	90.9%	100%
GRAVE	90.9%	90.9%	81.8%
PAUSA	100%	100%	100%
AVANCE	100%	100%	100%
SIGA	100%	100%	100%
VOLTE	100%	100%	100%
EJETE	100%	100%	100%
DESLIGA	90.9%	100%	90.9%
APAGUE	100%	100%	100%
TEMPO DE PROCESSAMENTO	**297hs : 53 min : 32.42s**	**63hs : 22min : 54.9s**	**106hs : 13 min : 28.54s**

TABELA 4: Número de comandos reconhecidos para as 11 locuções do comando testado pelos *modelos: "Backpropagation" / "Radial Basis"* / "PSHNN" que não obtiveram 100% de acerto no reconhecimento.

LOCUÇÕES	PALAVRAS TESTADAS – 11 –			
Foram reconhecidas como: * /* /*	**liga**	**pare**	**grave**	**desliga**
liga	9 / 11 / 10		1 / - / 1	
pare		10 / 10 / 11	1 / 1 / -	
grave		- / 1 / -		10 /10 / 9
pausa		1 / - / -		
apague			- / - / 2	
desliga	2 / - / 1			10 /11/10

**modelo "backpropagation"* / *modelo *radial basis* / *modelo *pshnn*

No modelo de rede *"pshnn"* utilizou-se no 1º e no 2º estágio uma rede MLP simples com algoritmo *"backpropagation"*, com treinamento *"batch"*, taxa de aprendizagem adaptativa, momento igual a 0.75 – valor empírico – e função de propagação *"logística sigmoidal"* nas 3 camadas. A configuração usada foi a mesma da *rede "backpropagation"* já descrita, ou seja, 481 x 69 x 20 x 10. Para o treinamento do 1º estágio foram utilizados 1000 padrões de entrada, 100 para cada um dos 10 comandos considerados, sendo rejeitados 308 padrões dentre os quais: 99 do comando **avance**, 11 do comando **siga**, 1 do comando **volte,** 99 do comando **desliga** e 98 do comando **apague,** tendo os seguintes limiares para aceitação dos vetores de entrada da rede *"pshnn"*:

limiar inferior ⇒	0.5	0.5	0.3012	0.5	0.2	0.2478	0.5	0.5	0.4726	0.2
limiar superior ⇒	*0.5*	*0.5*	0.5	0.5	0.5308	0.7272	*0.5*	*0.5*	0.8	0.6690

O treinamento do 1º estágio levou 13056 *"epochs"*, alcançando erro de 137,8170, tendo o tempo decorrido de 98hs:21min: 20.66seg. Para o treinamento do 2º estágio, a entrada da rede será um vetor de 481 x 308, ou seja, 481 características com os 308 padrões de entrada que foram rejeitados no 1º estágio da rede, sendo que este vetor é passado por uma transformação linear – com base no conjunto de autovetores – de forma a fazer uma rotação em relação aos eixos ortogonais, de maneira que torne mais fácil sua classificação neste estágio. O 2º estágio da rede foi construído com a mesma configuração adotada no 1º estágio e seus limiares de aceitação finalizaram o treinamento com os seguintes valores:

limiar inferior ⇒ 0.5 0.5 0.5 0.5 0.5 0.5 0.2 0.5 0.5 0.5

limiar inferior ⇒ 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5.

O treinamento deste 2º estágio levou 3200 *"epochs"*, alcançando erro de 1,0168, tendo o tempo decorrido de 7hs:52min:7.8seg. Portanto o treinamento global da rede *"pshnn"* levou 106hs:13min:28.54seg para um total de 16256 *"epochs"*. Para teste foram utilizados 110 padrões de entrada, 11 para cada comando, sendo que três de locutores que participaram do treinamento. No 1º estágio da rede foram rejeitados 35 padrões, sendo 1 do comando **liga**, 2 do comando **grave**, 11 do comando **avance**, 10 do comando **desliga**, e 11 do comando **apague**, enquanto que no 2º estágio não foi rejeitado nenhum padrão. O uso da transformação linear – realizada no treinamento – possibilitou que vetores que foram rejeitados no 1º estágio fossem classificados no 2º, como por exemplo as locuções dos comandos liga, grave, avance, desliga e apague, que apresentaram a seguinte saída no 1º estágio – tabela 5 –, sendo o destaque em negrito o valor da saída desejada – valor próximo de 1 –, o destaque sublinhado duas vezes o(s) valor(es) máximo(s) da saída obtido pela rede e o destaque em negrito sublinhado o valor da saída após a transformação – 2º estágio.

Na tabela 3 observa-se o resultado final obtido no teste – 1º e 2º estágios – em porcentagem, onde os comandos **pare, pausa, avance, siga, volte, ejete** e **apague** apresentaram taxa de acerto de 100% – 11 locuções enquanto que os comandos **liga, grave e desliga** apresentaram 90.9% – 10 locuções –, 81.8% – 9 locuções e 90.9% de acerto, respectivamente. Na tabela 4 são apresentados os resultados do teste – em número de locuções reconhecida para o comando testado - que obtiveram taxa de acerto menor que 100 %, onde o 3º resultado é para o modelo *"pshnn"*. Como exemplo, pode-se ver na tabela 4 que das 11 locuções testadas do comando **liga**, 10 foram reconhecidas corretamente, enquanto que 1 foi reconhecida como **desliga**.

AVALIAÇÃO DOS RESULTADOS

Com base nos resultados apresentados, observa-se os comandos **pausa, avance, siga, volte, ejete** e **apague** apresentaram taxa de acerto de 100% para os três tipos de rede estudados, mostrando portanto que são comandos "robustos" e de fácil reconhecimento para uma aplicação prática. Já os comandos **liga** – /l/ – alveolar (surda) –, **desliga** - /d/ - línguodental (sonora) –, **pare** – /p/ – oclusiva (surda) – e **grave** – /g/ – oclusiva (sonora) –, são os que geralmente apresentam problemas. Isto é devido ao fato das locuções serem foneticamente parecidas, o que nos leva a dizer que as características utilizadas não foram capazes de distinguir tão bem estas locuções quanto as das outras. Será, pois, necessário fazer novos testes adicionando características, de forma a tentar aumentar a distinção entre palavras parecidas. O comando **grave** apresentou erro nas 3 redes testadas, sendo sempre na locução 1022p3r1, o que nos leva a dizer que o problema esta na locução.

Convém aqui ressaltar, que a configuração utilizada nos 3 modelos de redes neurais foi a que apresentou melhor resultado dentro dos diversos experimentos executados em labo-

ratório, onde foram feitos testes com várias configurações diferentes de uma, duas e três camadas.

TABELA 5: Resultados obtidos na rede PSHNN antes e após a transformação, sendo: x.xx o valor da saída desejada, x.xx o(s) valor(es) máximo(s) da saída obtido pela rede e **x.xx** o valor da saída após a transformação – 2º estágio.

Vocabulário	**Saída da rede**									
LIGA	**0.28**	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.60	0.00
	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	**0.99**	0.00
GRAVE	0.00	0.32	**0.01**	0.00	0.19	0.00	0.00	0.00	0.00	0.08
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.99**
AVANCE	0.00	0,01	0.00	0.00	**0.43**	0.06	0.00	0.00	0.00	0.50
	0.00	0.00	0.00	0.00	**0.99**	0.01	0.00	0.00	0.00	0.00
DESLIGA	0.00	0.00	0.00	0.01	0.00	0.52	0.00	0.01	**0.83**	0.00
	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	**0.99**	0.00
APAGUE	0.00	0.01	0.00	0.00	0.44	0.07	0.00	0.00	0.00	**0.53**
	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.60	**0.99**

CONCLUSÕES

Através de diversos experimentos notou-se que a determinação exata dos pontos extremos nas locuções é de fundamental importância para o desempenho final do sistema RPI, assim como a pronúncia correta e bem clara da palavra a ser treinada e testada.

Verificou-se que as diferenças básicas de um modelo em relação ao outro são: o custo computacional e o tempo de processamento, muito menores na rede *"radial basis"*. Outro, aspecto relevante a considerar é o fato do modelo *"radial basis"*, no 1º estágio, poder ser utilizado ainda para avaliar a relevância de cada característica do sinal de voz, além de permitir trabalhar-se de forma paralela no treinamento deste estágio, ou seja, de forma independente, podendo, assim, reduzir ainda mais o tempo de treinamento. Esta rede também mostrou-se ser robusta, visto que a utilização de 500 padrões de entrada para treinamento apresentou resultado semelhante ao obtido com 1000 padrões, com base na utilização das 110 locuções de teste.

Quanto a comparação dos 3 modelos de redes neurais estudados: *"backpropagation"*, *"radial basis"* e *"pshnn"*, verificou-se que o desempenho alcançado foi quase equivalente no que toca ao nível de acerto, sendo a taxa média deste: 95.5%, 98.18% e 96.36%, respectivamente. Logo, com base nestes resultados, pode-se dizer que o sistema é viável para aplicações que visem o reconhecimento de comandos isolados.

Finalmente, verificou-se que os protótipos implementados mostraram-se eficientes e robustos, visto que, tendo-se utilizado apenas locutores masculinos na fase de treinamento das 3 redes neurais, estas ao serem testadas com algumas locuções de vozes femininas, conseguiram reconhecer os comandos com uma taxa muito boa de acerto.

BIBLIOGRAFIA

1. BEZERRA, Marconi dos Reis, Reconhecimento Automático de Locutor para Fins Forenses, Utilizando Técnicas de Redes Neurais, IME - Tese de Mestrado, Dezembro 1994.
2. DUDA, 0. Richard & HART, E. Peter, Pattern Classification and Scene Analysis, Wiley-Interscience, 1973, pp. 114-118.
3. RABINER, L. R. & SHAFER, R. W., Digital Processing of Speech Signals, Prentice-Hall, Inc., 1978.
4. RABINER, Lawrence & JUANG, Biing-Hwang, Fundamentals of Specch Recognation, Prentice-Hail, Inc., 1993.
5. DINIZ, Suelaine dos Santos, Uso de Técnicas Neurais para o Reconhecimento de Comandos a Voz, IME - Tese de Mestrado, Julho 1997.